ALTAIR

USER GUIDE

Altair Embed®/eDrives 2025.2

Creating Electric Motor Drive Firmware

ALTAIR © Altair Engineering, Inc. All Rights Reserved. / altair.com / Contact Us

Intellectual Property Rights Notice:

Copyright ©1986-2025 Altair Engineering Inc. All Rights Reserved.

This Intellectual Property Rights Notice is exemplary, and therefore not exhaustive, of the intellectual property rights held by Altair Engineering Inc. or its affiliates. Software, other products, and materials of Altair Engineering Inc. or its affiliates are protected under laws of the United States and laws of other jurisdictions.

In addition to intellectual property rights indicated herein, such software, other products, and materials of Altair Engineering Inc. or its affiliates may be further protected by patents, additional copyrights, additional trademarks, trade secrets, and additional other intellectual property rights. For avoidance of doubt, copyright notice does not imply publication. Copyrights in the below are held by Altair Engineering Inc. or its affiliates. Additionally, all non-Altair marks are the property of their respective owners. If you have any questions regarding trademarks or registrations, please contact marketing and legal.

This Intellectual Property Rights Notice does not give you any right to any product, such as software, or underlying intellectual property rights of Altair Engineering Inc. or its affiliates. Usage, for example, of software of Altair Engineering Inc. or its affiliates is governed by and dependent on a valid license agreement.

Altair® HyperWorks®, a Design & Simulation Platform
Altair® AcuSolve® ©1997-2025
Altair® Activate® ©1989-2025
Altair® Automated Reporting Director™ ©2008-2022
Altair® Battery Damage Identifier™ ©2019-2025
Altair® CFD™ ©1990-2025
Altair Compose® ©2007-2025
Altair® ConnectMe™ ©2014-2025
Altair® DesignAI™ ©2022-2025
Altair® DSim® ©2024-2025
Altair® DSim® Cloud ©2024-2025
Altair® DSim® Cloud CLI ©2024-2025
Altair® DSim® Studio ©2024-2025
Altair® EDEM™ ©2005-2025
Altair® EEvision™ ©2018-2025
Altair® ElectroFlo™ ©1992-2025
Altair Embed® ©1989-2025
Altair Embed® SE ©1989-2025
Altair Embed®/Digital Power Designer ©2012-2025
Altair Embed®/eDrives ©2012-2025
Altair Embed® Viewer ©1996-2025
Altair® e-Motor Director™ ©2019-2025
Altair® ESAComp® ©1992-2025
Altair® expertAI™ ©2020-2025
Altair® Feko® ©1999-2025
Altair® FlightStream® ©2017-2025
Altair® Flow Simulator™ ©2016-2025
Altair® Flux® ©1983-2025
Altair® FluxMotor® ©2017-2025
Altair® GateVision PRO™ ©2002-2025
Altair® Geomechanics Director™ ©2011-2022
Altair® HyperCrash® ©2001-2023
Altair® HyperGraph® ©1995-2025
Altair® HyperLife® ©1990-2025
Altair® HyperMesh® ©1990-2025
Altair® HyperMesh® CFD ©1990-2025
Altair® HyperMesh ® NVH ©1990-2025
Altair® HyperSpice™ ©2017-2025
Altair® HyperStudy® ©1999-2025
Altair® HyperView® ©1999-2025

Altair® HyperView Player® ©2022-2025
Altair® HyperWorks® ©1990-2025
Altair® HyperWorks® Design Explorer ©1990-2025
Altair® HyperXtrude® ©1999-2025
Altair® Impact Simulation Director™ ©2010-2022
Altair® Inspire™ ©2009-2025
Altair® Inspire™ Cast ©2011-2025
Altair® Inspire™ Extrude Metal ©1996-2025
Altair® Inspire™ Extrude Polymer ©1996-2025
Altair® Inspire™ Form ©1998-2025
Altair® Inspire™ Mold ©2009-2025
Altair® Inspire™ PolyFoam ©2009-2025
Altair® Inspire™ Print3D ©2021-2025
Altair® Inspire™ Render ©1993-2025
Altair® Inspire™ Studio ©1993-2025
Altair® Material Data Center™ ©2019-2025
Altair® Material Modeler™ ©2019-2025
Altair® Model Mesher Director™ ©2010-2025
Altair® MotionSolve® ©2002-2025
Altair® MotionView® ©1993-2025
Altair® Multi-Disciplinary Optimization Director™ ©2012-2025
Altair® Multiscale Designer® ©2011-2025
Altair® newFASANT™©2010-2020
Altair® nanoFluidX® ©2013-2025
Altair® NLView™ ©2018-2025
Altair® NVH Director™ ©2010-2025
Altair® NVH Full Vehicle™ ©2022-2025
Altair® NVH Standard™ ©2022-2025
Altair® OmniV™ ©2015-2025
Altair® OptiStruct® ©1996-2025
Altair® PhysicsAI™ ©2021-2025
Altair® PollEx™ ©2003-2025
Altair® PollEx™ for ECAD ©2003-2025
Altair® PSIM™ ©1994-2025
Altair® Pulse™ ©2020-2025
Altair® Radioss® ©1986-2025
Altair® romAI™ ©2022-2025
Altair® RTLvision PRO™ ©2002-2025
Altair® S-CALC™ ©1995-2025
Altair® S-CONCRETE™ ©1995-2025
Altair® S-FRAME® ©1995-2025
Altair® S-FOUNDATION™ ©1995-2025
Altair® S-LINE™ ©1995-2025
Altair® S-PAD™ © 1995-2025
Altair® S-STEEL™ ©1995-2025
Altair® S-TIMBER™ ©1995-2025
Altair® S-VIEW™ ©1995-2025
Altair® SEAM® ©1985-2025
Altair® shapeAI™ ©2021-2025
Altair® signalAI™ ©2020-2025
Altair® Silicon Debug Tools™ ©2018-2025
Altair® SimLab® ©2004-2025
Altair® SimLab® ST ©2019-2025
Altair® SimSolid® ©2015-2025
Altair® SpiceVision PRO™ ©2002-2025
Altair® Squeak and Rattle Director™ ©2012-2025
Altair® StarVision PRO™ ©2002-2025
Altair® Structural Office™ ©2022-2025

Altair® Sulis™©2018-2025
Altair® Twin Activate® ©1989-2025
Altair® UDE™ ©2015-2025
Altair® ultraFluidX® ©2010-2025
Altair® Virtual Gauge Director™ ©2012-2025
Altair® Virtual Wind Tunnel™ ©2012-2025
Altair® Weight Analytics™ ©2013-2022
Altair® Weld Certification Director™ ©2014-2025
Altair® WinProp™ ©2000-2025
Altair® WRAP™ ©1998-2025

Altair® HPCWorks®, a HPC & Cloud Platform
Altair® Allocator™ ©1995-2025
Altair® Access™ ©2008-2025
Altair® Accelerator™ ©1995-2025
Altair® Accelerator™ Plus ©1995-2025
Altair® Breeze™ ©2022-2025
Altair® Cassini™ ©2015-2025
Altair® Control™ ©2008-2025
Altair® Desktop Software Usage Analytics™ (DSUA) ©2022-2025
Altair® FlowTracer™ ©1995-2025
Altair® Grid Engine® ©2001, 2011-2025
Altair® InsightPro™ ©2023-2025
Altair® InsightPro™ for License Analytics ©2023-2025
Altair® Hero™ ©1995-2025
Altair® Liquid Scheduling™ ©2023-2025
Altair® Mistral™ ©2022-2025
Altair® Monitor™ ©1995-2025
Altair® NavOps® ©2022-2025
Altair® PBS Professional® ©1994-2025
Altair® PBS Works™ ©2022-2025
Altair® Simulation Cloud Suite (SCS) ©2024-2025
Altair® Software Asset Optimization (SAO) ©2007-2025
Altair® Unlimited™ ©2022-2025
Altair® Unlimited Data Analytics Appliance™ ©2022-2025
Altair® Unlimited Virtual Appliance™ ©2022-2025

Altair® RapidMiner®, a Data Analytics & AI Platform
Altair® AI Hub ©2023-2025
Altair® AI Edge™ ©2023-2025
Altair® AI Cloud ©2022-2025
Altair® AI Studio ©2023-2025
Altair® Analytics Workbench™ ©2002-2025
Altair® Graph Lakehouse™ ©2013-2025
Altair® Graph Studio™ ©2007-2025
Altair® Knowledge Hub™ ©2017-2025
Altair® Knowledge Studio® ©1994-2025
Altair® Knowledge Studio® for Apache Spark ©1994-2025
Altair® Knowledge Seeker™ ©1994-2025
Altair® IoT Studio™ ©2002-2025
Altair® Monarch® ©1996-2025
Altair® Monarch® Classic ©1996-2025
Altair® Monarch® Complete™ ©1996-2025
Altair® Monarch® Data Prep Studio ©2015-2025
Altair® Monarch Server™ ©1996-2025
Altair® Panopticon™ ©2004-2025

Altair® Panopticon™ BI ©2011-2025
Altair® SLC™ ©2002-2025
Altair® SLC Hub™ ©2002-2025
Altair® SmartWorks™ ©2002-2025
Altair® RapidMiner® ©2001-2025

Altair One® ©1994-2025
Altair® CoPilot™ ©2023-2025
Altair® Drive™ ©2023-2025
Altair® License Utility™ ©2010-2025
Altair® TheaRender® ©2010-2025
OpenMatrix™ ©2007-2025
OpenPBS® ©1994-2025
OpenRadioss™ ©1986-2025

Acknowledgments

Prof. Dr. Ir. Duco W. J. Pulle
University professor RWTH-ISEA Germany
Chief Electrical Drive Consultant for Altair Embed

Prof. Dr. Peter Freere
University professor NMU South Africa

Peter Darnell
Chief Architect Model Based Embedded Tools Altair

USER GUIDE: version 28

Preface

The purpose of this 'Altair Embed®User Guide' is to provide technical support for the examples that are present in the Embed 'eDRIVES examples' pull-down menu. These examples cover a comprehensive range of electrical drive applications designed to reduce the development time needed for potential users to get their machine(s) spinning. In terms of 'machines', the guide covers the most commonly used topologies namely: Brushed DC , PM synchronous, Hybrid steppers, Brushless DC, IPM and three-phase/one-phase squirrel cage Induction machines. Furthermore, attention is also given to topologies such as: Hybrid steppers and Synchronous reluctance machines. Care has been taken to provide the necessary theoretical models in support of the machines that are used in the examples.
The 'term' electrical drive also reflects the fact that a substantial control and converter component must also be addressed. In this context, the GUIDE provides examples and a theoretical outline of key principles concerned with speed and current control, switching converters and field-oriented control, both sensored and sensorless. For many application examples, a development tool-chain is provided, which consists of a 'simulation' (SIM), 'processor in the loop'(PIL), 'PSIM co-simulation' (COSIM) and 'hardware in the loop' (HIL) examples, so that users are able to become familiar with all aspects of an application.

Contents

1 Introduction 11
1.1 Importance of Electrical Drives and Electrical Machines 12
1.2 Key components of the electrical drive system 12
1.3 Notational conventions 15
1.3.1 Voltage and current conventions 15
1.3.2 Mechanical conventions 15
1.4 Use of building blocks to represent equations 16
1.4.1 Fixed point versus floating point representation 18
1.4.2 Use of Scaling for fixed point formatted numbers 20
1.4.3 Continuous and discrete time operational issues 21
1.4.4 Basic Control block library 24

2 Drive principles and development 27
2.1 Control of Electrical Drives 27
2.1.1 Gimbal Drive example 28
2.1.2 Use of a Speed Control loop 29
2.1.3 Overview of DC and AC machines 33
2.1.3.1 Brushed DC 37
2.1.3.2 Non-salient PM: PMsyn and BLDC 39
2.1.3.3 Salient PM: IPM and hybrid stepper 42
2.1.3.4 Synchronous Reluctance Motor (SynRM) 45
2.1.3.5 Induction machines 47
2.1.4 Current control for electrical drives 54
2.1.5 Space Vector modulation and Timing Issues 60
2.1.6 eDRIVES Example structure 66

3 Brushed DC drives 73
3.1 Armature Voltage Control 73
3.1.1 Armature Voltage Control-TI 75
3.1.1.1 Voltage control-F069-boostXL8301: SIM 76
3.1.1.2 Voltage control-F069-boostXL8301: PIL 79

3.1.1.3 Voltage control-F069-boostXL8301:COSIM . . . 80
3.1.1.4 Voltage control-F069-boostXL8301: HIL 87
3.1.1.5 Voltage control-F0049-GANINV: HIL 94
3.1.2 Armature Voltage Control-STM 100
3.1.2.1 Voltage control-F303-ESC001: SIM 100
3.1.2.2 Voltage control-F303-ESC001: PIL 103
3.1.2.3 Voltage control-F303-ESC001:COSIM 104
3.1.2.4 Voltage control-F303-ESC001: HIL 108
3.2 Armature Current Control 114
3.2.1 Armature Current Control-TI 114
3.2.1.1 Current control-F069-boostXL8301: SIM 114
3.2.1.2 Current control-F069-boostXL8301: PIL 117
3.2.1.3 Current control-F069-boostXL8301:COSIM . . 118
3.2.1.4 Current control-F069-boostXL8301: HIL 121
3.2.1.5 Current control-F0049-GANINV: HIL 126
3.2.2 Armature Current Control-STM 131
3.2.2.1 Current control-F303-ESC001: SIM 131
3.2.2.2 Current control-F303-ESC001: PIL 134
3.2.2.3 Current control-F303-ESC001:COSIM 135
3.2.2.4 Current control-F303-ESC001: HIL 138
3.3 Armature Speed Control 144
3.3.1 Armature Speed Control-TI 144
3.3.1.1 Speed control-F069-boostXL8301: SIM 144
3.3.1.2 Speed control-F069-boostXL8301: HIL 147

4 PMsyn-BLDC 155
4.1 Voltage & Frequency Control 155
4.1.1 Voltage & Frequency Control-TI 156
4.1.1.1 Voltage & Frequency control-F0049-GANINV: SIM 156
4.1.1.2 Voltage & Frequency control-F0049-GANINV: PIL 159
4.1.1.3 Voltage & Frequency control-F0049-GANINV:COSIM 161
4.1.1.4 Voltage & Frequency control-F0049-GANINV: HIL 169
4.1.1.5 Voltage & Frequency control-F069-GANINV: HIL 176
4.1.1.6 Voltage & Frequency control-F027-boostXL8301: HIL 183
4.1.1.7 Voltage & Frequency control-F069-DRV8301HC: HIL 189
4.1.1.8 Voltage & Frequency control-F069-DRV8312: HIL 195

4.1.1.9 Voltage & Frequency control-F28379-boostXL8305: HIL 201
4.1.1.10 Voltage & Frequency control-F069-boostXL8305: HIL 207
4.1.1.11 Voltage & Frequency control-F069-HVkit: HIL 213
4.1.1.12 Voltage & Frequency control-F0039-GANINV: HIL 218
4.1.2 Voltage & Frequency Control-STM 225
4.1.2.1 Voltage & Frequency control-F303-ESC001: SIM 225
4.1.2.2 Voltage & Frequency control-F303-ESC001: PIL 228
4.1.2.3 Voltage & Frequency control-F303-ESC001:COSIM 229
4.1.2.4 Voltage & Frequency control-F303-ESC001: HIL 234
4.1.2.5 Voltage & Frequency control-F405-VESC6-MKIII: HIL 241
4.1.2.6 Voltage & Frequency control-G473E-B-DRIVE: HIL 246
4.2 Current & Frequency Control 252
4.2.1 Current & Frequency Control-TI 253
4.2.1.1 Current & Frequency control-F0049-GANINV: SIM 254
4.2.1.2 Current & Frequency control-F0049-GANINV: PIL 258
4.2.1.3 Current & Frequency control-F0049-GANINV:COSIM 259
4.2.1.4 Current & Frequency control-F0049-GANINV: HIL 263
4.2.1.5 Current & Frequency control-Turnigy-F069-boostXL8301: HIL 268
4.2.2 Current & Frequency Control-STM 276
4.2.2.1 Current & Frequency control-F303-IHM07: SIM 276
4.2.2.2 Current & Frequency control-F303-IHM07: PIL 279
4.2.2.3 Current & Frequency control-F303-IHM07:COSIM 280
4.2.2.4 Current & Frequency control-F303-IHM07: HIL 284
4.2.2.5 Current & Frequency control-F303-ESC001: HIL 290
4.3 Sensored Field Oriented Control with encoder 296
4.3.1 Sensored Field Oriented Control & encoder-TI 297
4.3.1.1 Sensored FOC with Enc-F0049-GANINV: SIM 298
4.3.1.2 Sensored FOC with Enc-F0049-GANINV: PIL 304
4.3.1.3 Sensored FOC with Enc-F0049-GANINV:COSIM 306
4.3.1.4 Sensored FOC with Enc-F0049-GANINV: HIL 310

4.3.2 Sensored Field Oriented Control & encoder-STM 318
4.3.2.1 Sensored FOC with Enc-G473E-B-DRIVE: SIM318
4.3.2.2 Sensored FOC with Enc-G473E-B-DRIVE: HIL324
4.4 Sensored Field Oriented Control with Hall sensors 330
4.4.1 Sensored Field Oriented Control with Hall sensors-TI . 332
4.4.1.1 Sensored FOC with Hall sensors-F0049-GANINV:SIM 332
4.4.1.2 Sensored FOC with Hall sensors-F0049-GANINV: HIL 338
4.5 Sensored Field Oriented Control with Encoder and Field-weakening 344
4.5.1 Sensored Field Oriented Control with Encoder and Field-weakening-TI 347
4.5.1.1 Sensored FOC with Encoder and Field-weakening-F0049-GANINV: SIM 348
4.5.1.2 Sensored FOC with Encoder and Field-weakening-F0049-GANINV: HIL 353
4.5.1.3 Sensored FOC with Encoder and Field-weakening-F069-boostXL8301: HIL 359
4.6 Sensorless Field Oriented Control PMsyn & BLDC 366
4.6.1 Sensorless Field Oriented Control using AFT®-STM . 370
4.6.1.1 AFT FOC-F303-ESC001:SIM 370
4.6.1.2 AFT FOC-F303-ESC001:HIL 376
4.6.2 Sensorless Field Oriented Control using AFT®-TI 386
4.6.2.1 AFT FOC-F0039-GANINV:SIM 386
4.6.2.2 AFT FOC-F0039-GANINV:HIL 391
4.6.3 Sensorless Field Oriented Control using FAST®-TI ... 399
4.6.3.1 FAST FOC-F069-GANINV:PIL 399
4.6.3.2 FAST FOC-F069-GANINV:COSIM 405
4.6.3.3 FAST FOC-F069-GANINV:HIL 409
4.6.3.4 FAST FOC-F069-boostXL8301-drone: HIL ... 419
4.6.3.5 FAST FOC-F027-boostXL8301-Turnigy:HIL . 428
4.7 Sensorless Field Oriented Control PMsyn & BLDC with Field-weakening 436
4.7.1 Sensorless Field Oriented Control PMsyn & BLDC with Field-weakening-TI 437
4.7.1.1 FAST FOC-F069-GANINV with Field-weakening: PIL 437
4.7.1.2 FAST FOC-F069-GANINV with Field-weakening: HIL 441

5 Hybrid Stepper 449
5.1 Voltage & Frequency Control 450
5.1.1 Voltage & Frequency Control-TI 451

5.1.1.1 Voltage & Frequency control-Stepper-F069-GANINV: SIM 451
5.1.1.2 Voltage & Frequency control-Stepper-F069-GANINV: HIL 457
5.2 Sensorless Field Oriented Control of Hybrid Steppers 461
5.2.1 Sensorless Field Oriented Control of hybrid stepper using FAST®-TI 464
5.2.1.1 FAST FOC-Stepper-F069-GANINV:PIL 464
5.2.1.2 FAST FOC-Stepper-F069-GANINV:HIL 470

6 IM-3phase 485
6.1 Voltage-Frequency Control-IM 485
6.1.1 Voltage-Frequency Control-TI 489
6.1.1.1 Voltage-frequency Control-F0049-GANINV:SIM 489
6.1.1.2 Voltage-frequency Control-F0049-GANINV:HIL 495
6.2 Sensored Field Oriented Control-IM 501
6.2.1 IM Sensored Field Oriented Control-TI 503
6.2.1.1 IM Sensored Field Oriented Control: F0049-GANINV:SIM 503
6.2.1.2 IM Sensored FOC with Enc-F0049-GANINV: PIL 509
6.2.1.3 IM Sensored FOC with Enc-F0049-GANINV: COSIM 512
6.2.1.4 IM Sensored FOC with Enc-F0049-GANINV: HIL 516
6.2.1.5 IM Sensored FOC with Enc-F069-HV kit: HIL 524
6.3 Sensored Field Oriented Control-IM with Dynamic Field Control (DFC) 530
6.3.1 IM Sensored Field Oriented Control with DFC-TI 532
6.3.1.1 IM Sensored Field Oriented Control wth DFC: F0049-GANINV:SIM 533
6.3.1.2 IM Sensored FOC with DFC and Enc-F0049-GANINV: HIL 539
6.4 Sensorless Field Oriented Control three-phase IM 545
6.4.1 Sensorless Field Oriented Control of IM using FAST®-TI 548
6.4.1.1 IM FAST FOC-F069-GANINV:PIL 549
6.4.1.2 IM FAST FOC-F069-GANINV: COSIM 555
6.4.1.3 IM Sensorless FOC with FAST-F069-GANINV: HIL 558
6.4.1.4 IM Sensorless FOC with FAST-F069-8301HC: HIL 567
6.4.1.5 IM Sensorless FOC with FAST-F069-HVkit: HIL 575

7 IM-1phase 589
7.1 Mains connected-single phase IM 590
7.2 Voltage-Frequency Control-single phase IM 594
7.2.1 Voltage-Frequency Control-single phase IM-TI 595
7.2.1.1 Voltage-Frequency Control-F069-HVkit: SIM . 596
7.2.1.2 Voltage-Frequency Control-F069-HVkit: COSIM602
7.2.1.3 Voltage-Frequency Control-F069-HVkit:HIL . . 605
7.3 Sensorless Field Oriented Control single phase IM 611
7.3.1 Sensorless Field Oriented Control of single phase IM using FAST®-TI 614
7.3.1.1 IM Sph FAST FOC-F069-HVkit:PIL 614
7.3.1.2 IM Sph FAST FOC-F069-HVkit: COSIM 621
7.3.1.3 IM Sph FAST FOC-F069-HVkit: HIL 627

8 Dual drive sensored 637
8.1 Dual drive sensored DC-PM 640
8.1.1 Dual drive sensored DC-PM:SIM 643
8.1.1.1 DC-Voltage control, PM-Torque control: SIM . 645
8.1.1.2 DC-Speed control, PM-Torque control: SIM . . 647
8.1.1.3 PM-Speed control, DC-Torque control: SIM . . 648
8.1.2 Dual drive sensored DC-PM:HIL 650
8.1.2.1 DC-Voltage control, PM-FOC torque control: HIL 656
8.1.2.2 DC-Speed control, PM-FOC torque control: HIL 658
8.1.2.3 PM-FOC speed control, DC-Torque control: HIL 660
8.2 Dual drive sensored PM-PM 662
8.2.1 Dual drive sensored PM-PM: SIM 663
8.2.2 Dual drive sensored PM-PM: HIL 666
8.3 Dual drive sensored IM-PM 673
8.3.1 Dual drive sensored IM-PM: SIM 675
8.3.1.1 IM-Open loop C&F control, PM-FOC torque control: SIM 677
8.3.1.2 IM-FOC speed control, PM-FOC torque control: SIM 679
8.3.1.3 IM-FOC torque control, PM-FOC speed control: SIM 681
8.3.2 Dual drive sensored IM-PM: HIL 685
8.3.2.1 IM-Open loop C&F control, PM-FOC torque control: HIL 690
8.3.2.2 IM-FOC speed control, PM-FOC torque control: HIL 692
8.3.2.3 IM-FOC torque control, PM-FOC speed control: HIL 694

8.4 Dual drive sensorless Hybrid Stepper sensored PM 696
8.4.1 Dual drive sensorless Hybrid Stepper sensored PM: PIL 699
8.4.2 Dual drive sensorless Hybrid Stepper sensored PM: HIL 705

9 IPM 719
9.1 IPM Sensored Field Oriented Control with encoder 732
9.1.1 IPM sensored FOC with Enc: SIM 733
9.1.1.1 Results, IPM sensored FOC with Enc: SIM-linear 736
9.1.1.2 Results, IPM sensored FOC with Enc: SIM-saturation 740
9.1.1.3 Results, IPM sensored FOC with Enc: SIM-linear-saturation 741
9.1.2 IPM Sensored FOC with Enc: PIL 744
9.1.3 IPM Sensored FOC with Enc: COSIM 748
9.1.3.1 Results, IPM sensored FOC with Encoder . . . 751
9.2 IPM Sensorless Field Oriented Control 755
9.2.1 ISEA IPM sensorless FOC using FAST: PIL 758
9.2.2 FluxMotor-IPM sensorless FOC using FAST®: PIL-COSIM 766
9.2.2.1 Results, IPM sensorless FOC using FAST® . . 774

10 SynRM 779
10.1 SynRM sensored Field Oriented Control with encoder 784
10.1.1 SynRM sensored FOC with Enc: SIM 785
10.2 Sensorless Field Oriented Control of SynRM 791
10.2.1 Sensorless Field Oriented Control of SynRM using FAST®-TI 792
10.2.1.1 FAST FOC-SynRM-F069-DRV8305:PIL 792
10.2.1.2 FAST FOC-SynRM-F069-DRV8305:HIL 798

References 809

Chapter 1
Introduction

Prior to discussing the control of electrical drives, it is prudent to provide an introductory overview of the concepts and the hardware used in this GUIDE.[14] In addition, the notational conventions, both electrical and mechanical, are introduced together with the control blocks used to build drive controllers in successive chapters. Both, so called floating and fixed point control blocks are introduced, together with continuous and discrete time modules, given that an 'real-time embedded control' approach is used throughout this GUIDE. The term 'embedded control' refers to the approach where the user envisaged control algorithm is developed by way of 'real-time embedded control' software, which has a graphical interface that allows full representation of the algorithm as it needs to function under real-time conditions. This means that users inputs and outputs, as shown in figure 1.1, are generated by the micro controller unit (MCU) to allow proper functioning of the electrical drive, which must be controlled by the algorithm. The embedded control software 'automatically' generates the C-code which in turn generates the out-file that runs the MCU. Hence the user does not require 'C-code literacy' to develop and run control algorithms, as will be shown extensively in this GUIDE. The

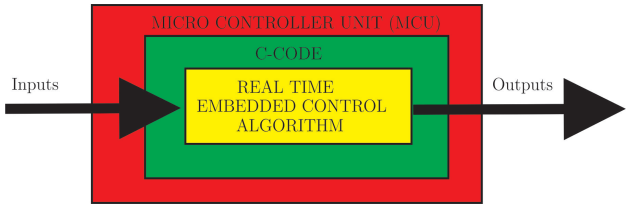


Fig. 1.1 Real-time embedded control concept

real-time embedded software used in the GUIDE is Altair Embed®[15]. Both so called floating and fixed point micro-controllers are available, hence the need to briefly discuss fixed point scaling principles in this chapter.

1.1 Importance of Electrical Drives and Electrical Machines

Electric motors are around us everywhere. Generators in power plants are connected to a three-phase power grid. Pumps in your heating system, refrigerator and vacuum cleaner are connected to a single phase AC grid and switched on or off by means of a simple contactor or electronic converter. In cars, a direct current (DC) battery is used for example, to provide power to the starter motor. This type of motor runs on direct current and in most cases is activated by a relay switch without any control.

Many applications driven by electric motors require more or less advanced control. Lowering the speed of a fan or pump can be considered a relatively simple task. Perhaps one of the most difficult ones is the dynamic positioning of a tug in a wafer-stepper application with nano-meter accuracy while accelerating at several g's. Another challenging controlled drive is an electric crane in a harbour that needs to be able to move an empty hook at high speed, navigate heavy loads up and down at moderate velocities and make a soft touchdown as close as possible to its intended final position. Other applications such as assembly robots, electric elevators, electric motor control in electric vehicles, trains, drones, or CD-players can, with regard to complexity, be situated somewhere in between.

Design and analysis of all electric drive systems requires not only knowledge of dynamic properties of different motor types, but also a good understanding of the way these motors interact with power-electronic converters and their loads. These power converters are used to control motor currents or voltages in various manners.

1.2 Key components of the electrical drive system

The drive concept of primary interest in this GUIDE is capable of what is referred to as 'adjustable speed' operation [12] [13] [28], which means that the machine can be made to operate over a wide speed range. A simplified structure of an adjustable speed drive is shown in figure 1.2, which makes use of a Texas Instruments®or ST-Micro®micro-controller unit (MCU). Various MCU types from both manufacturers will be used in this GUIDE for real-time embedded control. Both low and 'high' (meaning mains voltage levels) voltage electrical drives are introduced.
A brief description of the components is given below:

- Motor: the term 'AC' (see figure 1.2), refers to either two (known as 'single phase')/three-phase induction (asynchronous) or non-salient (PMSyn &

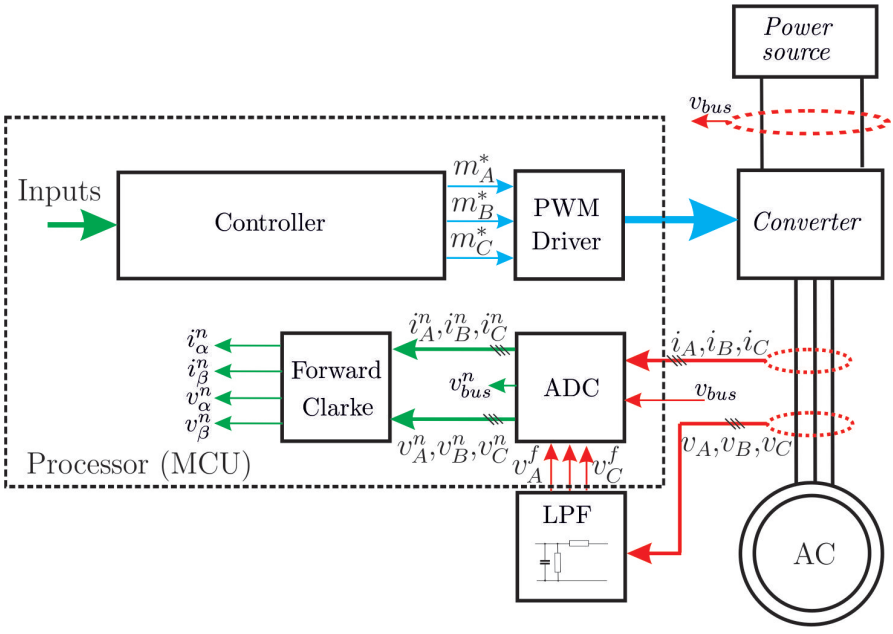


Fig. 1.2 Typical drive set-up

BLDC)/salient (IPM) permanent magnet synchronous machines, both low and 'mains' voltage levels. However, Synchronous Reluctance (SynRM), PM hybrid stepper, four-phase axial flux 'switched reluctance motor' (SRM) and brushed DC can also be connected to a three-phase converter, as is discussed in the GUIDE. Note that the term 'motor' refers to a machine which operates as a motor, i.e. energy flows from the motor to the load. When the energy flows in the opposite direction a machine is said to operate as a generator.

- Converter: This unit contains a set of power electronic (semiconductor) switches which are used to control the energy transfer between the power supply and motor. The use of switches is important given that no power is dissipated (in the ideal case) when the switches are either open or closed. Hence, theoretically, the efficiency of such a converter is 100%, which is important particularly for large converters given the fact that semiconductor devices cannot operate at high temperatures. Hence, it is not possible to absorb high losses which inevitably appear in the form of heat.
- Processor (MCU): This unit, also known as a 'micro controller unit' controls the switches of the converter module using a 'PWM Driver', which in turn is provided by the modulation index references m_A^*, m_B^*, m_C^*.
  The 'controller' module contains the control algorithm, as developed via

real-time embedded control software (see figure 1.1) and generates the reference modulation indices. Inputs to the controller depend on the functionality imposed by the user and ranges from: reference speed, reference current/voltage values, measured voltages/currents. The 'ADC' unit samples the measured currents i_A, i_B, i_C and the low-pass filtered (LPF) converter u_A^f, u_B^f, u_C^f and DC bus u_{bus} voltages. A 'Forward Clarke' unit [12] transforms the three-phase ADC voltages/currents to a two-phase, subscript α, β domain, as required for control purposes.
A substantial range of control algorithms will be discussed in this GUIDE and this includes sensorless operation, where voltage/current measurements are utilized to determine the position of the motor flux vector. For Texas Instruments MCU's, the 'FAST®algorithm is used, whereas for ST-Micro®an alternative control algorithm is deployed.

- Power supply: the converter requires, in our case, a 24V, 48V or 350V DC voltage source (DC voltage link), depending on the motor type in use. The power is obtained directly from a DC power supply source, as is the case in this GUIDE (for safety purposes).

Shown in figure 1.3, is a subset of drive hardware used in this GUIDE that meets the topology shown in figure 1.2. Furthermore, an effort has been made to ensure that the converters used in this GUIDE can indeed be purchased by the reader. The hardware setup given in figure 1.3, shows the diversity of

Fig. 1.3 Subset of Drive hardware used in this GUIDE

electrical drive applications covered. Shown are low/medium power converters from Texas Instruments®and ST-Micro®. The motor types indicated are: Synchronous Reluctance, PM 'outrunner', Induction, Hybrid stepper and PM synchronous, all of whom will be exposed to a plethora of control algorithms.

1.3 Notational conventions

1.3.1 Voltage and current conventions

The conventions used in this GUIDE for the voltage and current variables are shown with the aid of figure 1.4. The diagram shows the variables voltage u

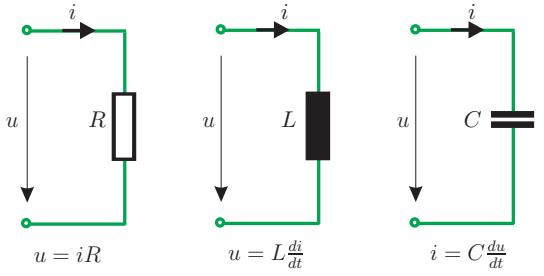


Fig. 1.4 Notation conventions used for electrical ideal components [12]

and current i, which are specifically given in 'lower case' notation, because they represent instantaneous values, i.e. are a function of time. The voltage and current 'arrows' shown in figure 1.4 point to the negative terminal of the respective circuit, i.e motoring arrow system, in which positive power $p = u\,i$ means power absorbed by the electrical circuit (load).

1.3.2 Mechanical conventions

The mechanical conventions used in this GUIDE are shown with the aid of figure 1.5. The electro-magnetic torque T_e produced by the machine corresponds with a power output $p_e = T_e\,\omega_m$, where ω_m represents the rotational speed, otherwise known as the angular frequency. The load torque T_L is linked to the power delivered to the load $p_L = T_l\,\omega_m$. Ignoring bearing and windage losses in the machine, the torque difference $T_e - T_l$ results in an acceleration $Jd\omega_m/dt$ of the total rotating mass, which is characterized by its inertia J . This rotating structure is represented as a lumped mass formed by the rotor

of the motor, motor shaft and load. The corresponding mechanical equation which governs this system is of the form

$$J\frac{d\omega_m}{dt} = T_e - T_l \tag{1.1}$$

The angular frequency may also be written as $\omega_m = d\theta/dt$ where θ represents the rotor angle. Figure 1.5 shows the machine operating as a motor,

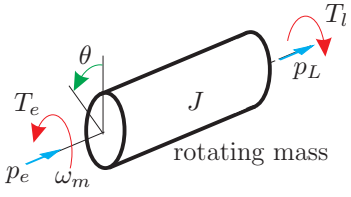


Fig. 1.5 Notation conventions used for mechanical quantities [12]

i.e. $T_e > 0$, $\omega_m > 0$. Note that motor operation also takes place when the conditions $T_e < 0$, $\omega_m < 0$ are satisfied. These motor conventions are used throughout this GUIDE.

1.4 Use of building blocks to represent equations

In the GUIDE use is made of Embed [15] for embedded control and simulation. The software ACTIVATE [17] and PSIM [16] is used for simulation, which are able to represent: electrical, mechanical or magnetic circuits with corresponding elements. For example, the mechanical system shown in figure 1.5 can be readily represented by a set of 'mechanical rotational' blocks, as may be observed from figure 1.6.
This type of approach can be extended to model complex rotation and/or linear mechanical systems, where sensors such as those shown for shaft speed and angle in the above example can be used for control purposes. When modelling electrical circuits using ACTIVATE or PSIM it is possible to also include power electronic components such as diodes and switching devices (such as IGBT's and MOSFET's). For the development of control structures, use is made of so called 'control blocks' to represent equations, in which case either ACTIVATE or Embed are deployed. This approach is also useful for representing an electrical circuit by one or more control blocks. Furthermore, discrete time control blocks can be combined to develop more complex 'compound' models, which can be directly used to generate a real time controller,

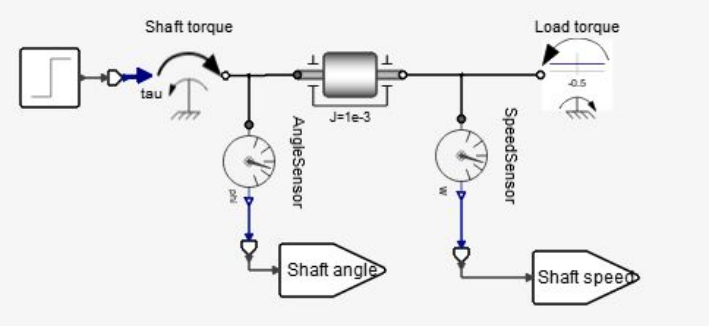


Fig. 1.6 ACTIVATE®representation of a lumped inertia system

without requiring any knowledge of C code, or the underlying code generation infra structure, as was discussed earlier (see figure 1.1).

Simulation models in this form can then be analysed by the reader in terms of the expected transient or steady-state response of an electrical drive. Furthermore, Embed®is used in combination with PSIM®or ACTIVATE®, for example, to examine the correct functioning of a fixed-point algorithm via simulation with a motor model and switching converter, prior to its use for real-time embedded control with the 'real' motor, as will become apparent in the next chapter.

An example of moving from electrical circuit to control block representation is given in figure 1.7.

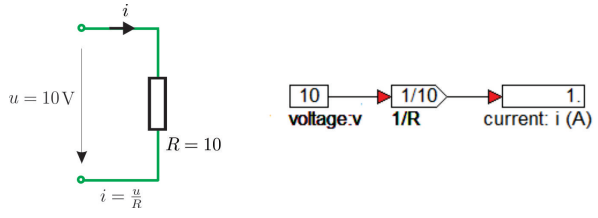


Fig. 1.7 Electrical circuit and EMBED based control block representations

The electrical circuit shown in figure 1.7 represents a resistance. This component has a known relation between voltage and current, as defined by Ohm's law: hence one can calculate current from voltage, voltage from current or resistance from both voltage and current. The control block approach assumes, in this case, that the voltage u, set to $10\,V$, is an input and the current i represents the output variable for this module known as a 'gain' module. The gain for this module must in this case be set to $1/R = 0.1$ for the chosen numerical example, in which case the current output will be equal to $i = 1.0\,A$

as shown on the numerical display. Note that the 'electrical circuit' model given in figure 1.7 can also be directly represented (as an electronic circuit component) using PSIM or ACTIVATE software. The same example can also be generated using the Embed 'fixed point' library, as shown in figure 1.8, which is used here because the intention is to generate control structures which can be experimentally verified, i.e implemented in real time.

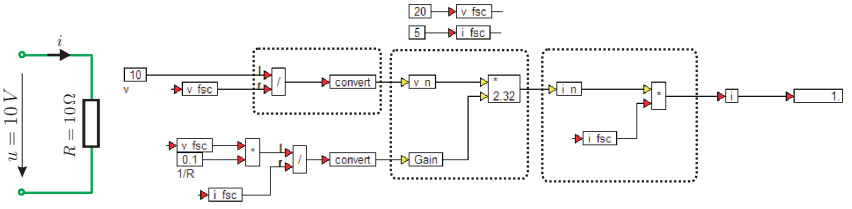


Fig. 1.8 Electrical and Embed based control block representations

Observation of figures 1.7 and 1.8 shows that the Embed control block implementations are considerably different. The fundamental reason for this is that the figure 1.7 version is implemented in so called 'floating point' and (for all practical purposes) in continuous time domain, whereas the Embed implementation is designed for use with a so called 'fixed point' MCU, which in turns implies the use of discrete time. Emphasis is placed here on the use of the Embed fixed point control block library, given that the latter will be used to build/develop 'embedded control' modules in this GUIDE. Prior to discussing the Embed implementation, it is therefore prudent to briefly discuss the underlying concept of fixed and floating point representation, as well as continuous and discrete time systems.

1.4.1 Fixed point versus floating point representation

Typically, microprocessor controllers, such as the one used in this GUIDE, operate with 32 bit word length, in which special significance is placed on the location of a series of bits relative to the radix point (like a 'decimal point'). In for example, floating point MCU's, the bit format as shown in equation (1.2) is used.

$$\mathrm{S}\underbrace{eeeeeeee}_{\text{8 bit exponent}}.\underbrace{fffffffffffffffffffffff}_{\text{23 bit quotient(fraction)}} \tag{1.2}$$

In this floating point format, the sign of the number is the 'S' bit, where 0 is allocated to positive numbers and 1 to negative numbers. The following eight 'e' bits represent the exponent of the floating point number, hence its maximum decimal value is 255. The quotient or fraction are the 23 'f' bits which are to the right of the radix point. Using this approach any number within a given range can be defined and the maximum value and resolution can be readily calculated, as is shown in tables (1.1),(1.2). Note that the maximum and minimum values for the floating point numbers are nearly equal, whereas this is different for a fixed point representation of a 32 bit number, as shown in equation (1.3).

$$\underbrace{S\,I}_{\text{2 bit Integer}} \cdot \underbrace{f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f\,f}_{\text{30 bit quotient(fraction)}} \tag{1.3}$$

Expression (1.3) shows a so called $2Q.30$ (normally simply referred to as 'IQ30') fixed point format, where the 30 represents the number of 'f' bits to the right of the radix point. This implies that (for a 32 bit word length) there are two remaining bits on the left side of the radix point, of which the 'S' bit is the sign, where 0 is allocated to positive numbers and 1 to negative numbers. The 'I' bit in this example is therefore one single bit. Hence the maximum positive value of the 'integer' part of the number is decimal '1'(integer bits $S = 0$, $I = 1$) and the minimum decimal value is -2 (integer bits $S = 1$, $I = 0$). In theory, the radix point can be moved to any location within the 32 bit word, but in practice it is sufficient to work with two formats namely $2Q.30$ (IQ30) and $8Q.24$ (IQ24), for reasons that will become apparent shortly. In the IQ24 format the radix point is moved to realize a 24 bit quotient or fraction. Accordingly the maximum, minimum and resolution values will also change as may be observed from tables 1.1 and 1.2.

format	max decimal value	min value
Floating point	$+2^{(255-127)}\,(1-2^{-23}) \approx +3.4028 * 10^{38}$	$-3.4028 * 10^{38}$
Fixed point: IQ30	$(2-2^{-30}) \approx 1.9999999991$	-2.000000000
Fixed point: IQ24	$(128-2^{-24}) \approx 127.9999999404$	-128.000000000

Table 1.1 Comparison of max/min values for 32 *bit* floating and fixed point representation

Observation of table 1.1 shows that the floating point format max/min values are large compared to the fixed point formats shown in said table. Moreover the max/min values are further reduced as the radix point is shifted to the

left, i.e. format change from IQ24 to IQ30. However the resolution increases as the radix point is moved to the left, as may be observed from table 1.2.

format	resolution
Floating point	$+2^{(0-127)}2^{-23} \approx 7.0065 * 10^{-45}$
Fixed point: IQ30	$2^{-30} \approx 9.3132 * 10^{-10}$
Fixed point: IQ24	$2^{-24} \approx 5.9605 * 10^{-8}$

Table 1.2 Comparison of resolution for 32 *bit* floating and fixed point representation

The floating point format is superior in terms of max/min range and have a fixed resolution. Historically, floating point MCU based controllers have been more expensive then fixed point controllers, where the resolution is dependent on word length and location of the radix point. Furthermore a fixed point controller algorithm can be executed on a floating point MCU, but not vice-versa. Hence in the GUIDE, for the sake of generality we will use fixed point formatting in either IQ30 or IQ24. Note that in the Embed models, the notation IQ30, IQ24 is replaced by 2.32, 8.32 respectively. In addition the sensorless control algorithms used in the laboratory sessions discussed in this GUIDE are implemented in fixed point, hence it is deemed important for the reader to be familiar with scaling concepts. Generally the IQ30 format is used where high resolution is deemed mandatory, whilst IQ24 is used when a larger (then IQ30) max/min range is required (in controllers with gain, for example).

1.4.2 Use of Scaling for fixed point formatted numbers

The use of fixed point formatted numbers, implies the need for 'scaling' variables to a per unit value, as will be demonstrated in this subsection. This scaling process will be expanded, as needed for each of the laboratory sessions, as will be come apparent shortly. In this case the scaling will be applied to the resistance example show in figure 1.8.

Basically scaling implies the allocation of a 'full scale' value to a given variable. For example, voltage u and current i variables may be written as shown in equation (1.4).

$$u = u_{\mathrm{n}}\, u_{\mathrm{fs}} \tag{1.4a}$$

$$i = i_{\mathrm{n}}\, i_{\mathrm{fs}} \tag{1.4b}$$

where the variables u_n, i_n represent the 'per unit' or scaled voltage and current values respectively. The 'fullscale' voltage and current value are defined as u_{fs}, i_{fs} respectively. In this example the expression $i = {}^{u}/_{R}$ needs to be scaled, which upon use of equation (1.4) gives

$$i_{\mathrm{n}} = u_{\mathrm{n}} \underbrace{\left(\frac{u_{\mathrm{fs}}}{R\, i_{\mathrm{fs}}} \right)}_{\text{gain}} \tag{1.5}$$

Shown in equation (1.5) is a 'gain' factor, which is precisely the value of the 'Gain' variable shown in figure 1.8. The input and output variable shown in the same figure represent the voltage and current respectively and accordingly these must be appropriately scaled using equation (1.4) in the 'pre-scaling' and 'post-scaling modules'. Pre-scaling implies that the input variable is multiplied by the inverse of the full scale value, whilst post-scaling requires multiplication of the per unit output variable by its full scale value. Note that a 'convert' module is used on the inputs to the 'fixed point' module shown in figure 1.8, which converts the floating point number to the designated fixed point (in this case IQ30) format.

1.4.3 Continuous and discrete time operational issues

For the purpose of explaining continuous time models it is helpful to consider an integrator circuit example as given in figure 1.9.

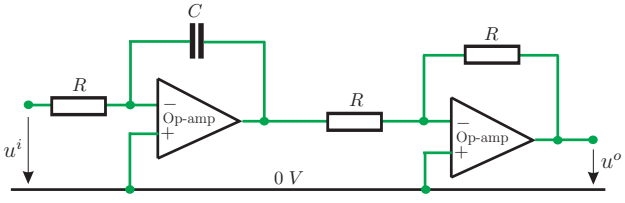


Fig. 1.9 Integrator example: continuous time domain

The relationship between output and input may be written as

$$u^o = \underbrace{\left(\frac{1}{RC} \right)}_{\text{gain}} \int u^i dt \tag{1.6}$$

where the signals u^o, u^i are assumed to be continuous, i.e have a value which is defined at any instance in time.

The controller is in our case digital, which means that the analog input variable, here in the form of the integrator input voltage $u^i(t)$, must be converted to a digital form using an 'analog-digital' (A/D) converter. The function of the unit is readily shown with the aid of figure 1.10.

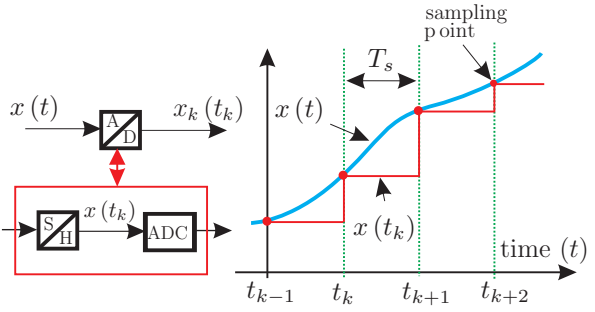


Fig. 1.10 A/D converter unit with example input/output waveforms

Figure 1.10 shows an analogue input function $x(t)$ (often a voltage) to the A/D converter. In reality this module consist of two sub-modules (as indicated in figure 1.10) namely a sample & hold (S/H) module and an analog to digital (ADC) converter, which converts the analogue input variable $x(t_k)$ to a digital representation $x_k(t_k)$.

The diagram shows an example waveform together with a set of discrete time points t_{k-1}, t_k, t_{k+1} where k can be any integer value. The difference in time between any two time points is constant and equal to the 'sampling interval period' T_s. The output of the S/H module is such that at these time points the input is 'sampled' i.e. the output is then set to equal the input value. This output value then stays constant untill the next sampling instance. Hence, at these 'sampling points' the output changes to match the instantaneous value found at the input of the converter. The output is therefore held constant during the sampling time. For example, the output $x(t_k)$ represents the value of the input variable as sampled at the time mark t_k. Subsequently these variables are then converted to digital values $x_k(t_k)$. In subsequent discussions the subscript k (for the variable x) will be dropped with the implicit understanding that the variables under consideration represent digital numbers.

The A/D units are used to sample the measured currents and voltages and these are then used by the MCU to calculate an output variable. In this example the continuous integrator needs to be remodeled for discrete, fixed point operation, so that it can be implemented in Embed. For this purpose it is helpful to undertake a first order approximation of equation (1.6) which gives

$$u^o(t_{k+1}) \cong \underbrace{\left(\frac{T_s}{RC}\right)}_{\text{gain}_\text{D}} u^i(t_k) + u^o(t_k) \tag{1.7}$$

The final step in realizing a fixed point implementation of equation (1.7) is scaling. Hence use of equation (1.4), which implies multiplication of expression (1.7) by the inverse of the full scale voltage which gives

$$u^o_\text{n}(t_{k+1}) \cong \underbrace{\left(\frac{T_s}{RC}\right)}_{\text{gain}_\text{D}} u^i_\text{n}(t_k) + u^o_\text{n}(t_k) \tag{1.8}$$

Note that the only difference between expressions (1.7),(1.8) is the use of per unit values, instead of floating point variables. An example, as given in figure 1.11, shows an implementation of equation (1.8).

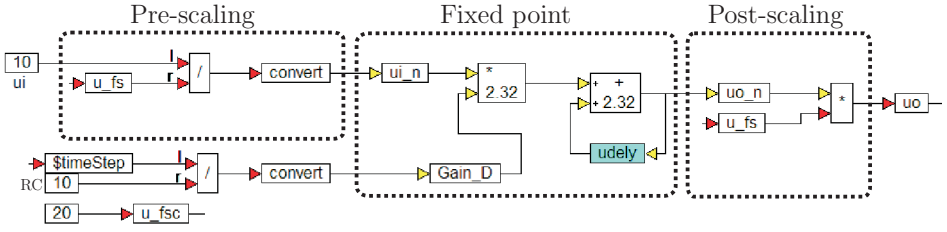


Fig. 1.11 Discrete, fixed point integrator example using Embed

In this example, the 'fixed' point module, implemented in IQ30, contains the discrete fixed point integrator , with gain module `Gain_D`, which is found using the gain value shown in equation (1.8). A one sample delay module `udely` is used to generate the $u^o_\text{n}(t_k)$ signal from $u^o_\text{n}(t_{k+1})$.
The simulation example was run over a period of 1 second, with a sampling time $T_s = 1\,\text{ms}$, a value which is placed in the *timestep* variable. Input to the integrator is a 10 V constant voltage, hence the theoretical integrator output value according to equation (1.6) should be 1V, given that product of the resistance and capacitor values `RC` was set to 10. Pre/Post scaling modules are again used for the conversion between per unit and non-scaled variables.

1.4.4 Basic Control block library

Throughout this GUIDE additional control blocks will be introduced as they are required. At this point, a basic set will be given which will form the basis for the first set of models to be discussed in this GUIDE. In each case the Embed floating and fixed point equivalent in per unit, IQ30 format will be shown. The Pre- and Post-Scaling module are not shown, to aid readability.

The first set of building blocks, as given in figure 1.12, are linked to 'example' transfer functions (also shown in said figure). Note for clarity that the left column of examples are in the continuous time domain, whereas those in the right column are discrete (sampled) .

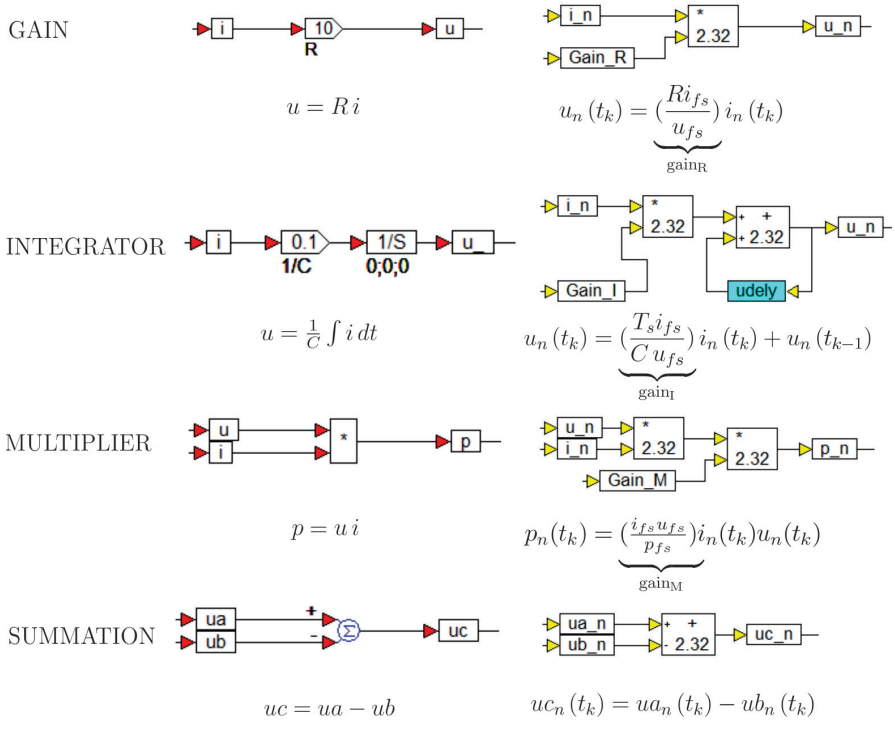


Fig. 1.12 Basic building block set

For example, the GAIN module has as input the current i and as output u. The gain is set to R and in the discrete version said gain is implemented using a multiplication module and input variable `Gain_R`. The INTEGRATOR example module has as input the variable i and output u whilst the gain of the integrator is $1/C$. The corresponding discrete gain `Gain_I` for the Embed module will be a function of the sampling time T_s used as discussed previously. When multiplying two variables in the time domain, a MULTIPLIER mod-

ule is used. This module differs from the given GAIN module in that the latter is used to multiply a variable with a constant. In the discrete version an additional `gain_M` factor is present, in the event that the full scale power value p_{fs} differs from the full scale voltage and current product $ufs\,i_{\mathrm{fs}}$ Finally, an example of a SUMMATION module is given. In this case the output is a variable uc and subtracts the input variable ub from input variable ua.

An example of combining some of these modules is readily given by considering the equation (1.9)

$$u = iR + L\frac{di}{dt} \tag{1.9}$$

which represents the voltage across a series network in the form of an inductance L and resistance R. To build a floating point control block representation, with the voltage as input variable and current as output variable, it is helpful to rewrite the expression in its differential equation form

$$\frac{di}{dt} = \frac{1}{L}\left(u - iR\right) \tag{1.10}$$

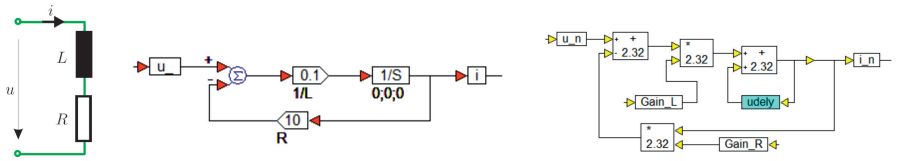
When integrated the output of the integrator is the variable i and the input of the integrator is given as $(u - iR)$, hence

$$i = \frac{1}{L}\int \left(u - iR\right) dt + i(0) \tag{1.11}$$

The initial current is assumed to be zero, i.e. $i\,(0) = 0$. An observation of equation (1.11) shows that the integrator input is formed by the input variable u from which the term iR must be subtracted. Use is made of a summation unit, as shown in the Embed floating point model presented in figure 1.13. The gain '$1/L$' present in equation (1.11) appears in the this model as a '$1/L$' gain as discussed previously. A discrete fixed point IQ30 format Embed implementation requires discretization and scaling of expression (1.11) as discussed in subsection 1.4.3. Subsequent mathematical handling of the equation gives;

$$i(t_k) \cong \underbrace{\left(\frac{T_s u_{\mathrm{fs}}}{L\, i_{\mathrm{fs}}}\right)}_{\mathrm{Gain_L}} \left(u_{\mathrm{n}}(t_k) - i_{\mathrm{n}}(t_k) \underbrace{\left(\frac{R i_{\mathrm{fs}}}{u_{\mathrm{fs}}}\right)}_{\mathrm{Gain_R}} \right) + i_{\mathrm{n}}(t_{k-1}) \tag{1.12}$$

which shows the presence of gains $\mathrm{Gain_R}$, $\mathrm{Gain_L}$, which also appear in the discrete Embed implementation shown in figure 1.13 as `Gain_R` and `Gain_L` respectively. This diagram also shows the electrical and floating point control block implementation for this example.
In PSIM or ACTIVATE this example can also be directly implemented using



Electrical Floating point Fixed point

Fig. 1.13 Example of using basic building blocks

two ‘electrical’ components that represent the inductance and resistance respectively.

Chapter 2
Drive Principles and Development

This chapter provides a compact overview of the underlying theoretical aspects of an electrical drive. The aim is to convey to the reader, with a minimum of mathematics, the key issues which are critical to controlling an electrical drive. In this context a fundamental understanding of AC machines is required in terms of their ability to generate torque. In the second part of this chapter the 'tool chain', otherwise referred to as the 'development approach' is discussed. Hence the steps used in this GUIDE move from a simulation model to an actual experimental laboratory drive, and are shown with the aid of hardware and software packages used in this GUIDE. More detailed analysis of the electrical principles and advanced control are given in our reference material [12] [13].

2.1 Control of Electrical Drives

This section outlines the basic control principles of any type of drive, starting from the need to achieve precise speed control. Consequently the basic control steps required to realize this task are outlined, starting from the speed controller, which in turn sets the torque level in a machine. Controlling torque means controlling the current, which in turn, requires the use of a current controller that makes use of a power electronic converter. The latter unit sets the discrete voltage levels in the connected AC machine. This logical chain of modules is discussed and the critical parameters that govern the speed and current controller gain and bandwidth are identified. Furthermore, a brief overview of the machines used in this GUIDE are given, in order to show how torque control can be achieved.

2.1.1 Gimbal Drive example

The gimbal motor setup (typically used to control the rotation of an object about an axis) shown in figure 2.1, is an example of a low cost electrical drive system, hence it is used here by means of illustration. The motor utilizes a 7 pole pair, low-speed 1000 rpm, three-phase permanent magnet 'outrunner' (named after the fact that the rotor is located on the outside of the stator) motor. For this example a cable is connected between the motor and the DRV8301-027-KIT . The unit, which houses the converter and the Texas Instruments F28027 control card, which contains a field-oriented, sensorless control algorithm, known as InstaSPIN [23], to be discussed at a later stage.

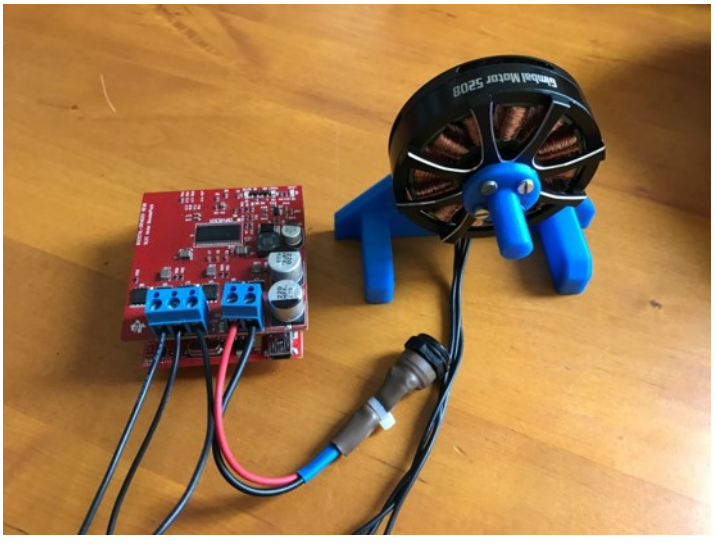


Fig. 2.1 Gimbal Drive example

The laptop, connected to the motor kit via a USB cable (not shown), runs the Embed software used to compile the C-code that accesses the InstaSPIN algorithm located in the ROM of the processor. This algorithm provides all the functionality to realize sensorless speed control of the motor and identification of the motor parameters. Power for the converter is provided by a 24 V supply (not shown). A more detailed overview of the drive itself, as given in figure 2.2, shows the PM motor, which is connected to the three-phase converter. In each of the converter legs a resistance is located between the 0 V bus and the lower switches, in order to measure the phase currents. In addition a 'voltage measurement unit' is used to filter and attenuate the phase voltages. Both voltages and currents are fed to an ADC unit inside the F28027 board, which is part of the Texas Instruments C2000 MCU family.

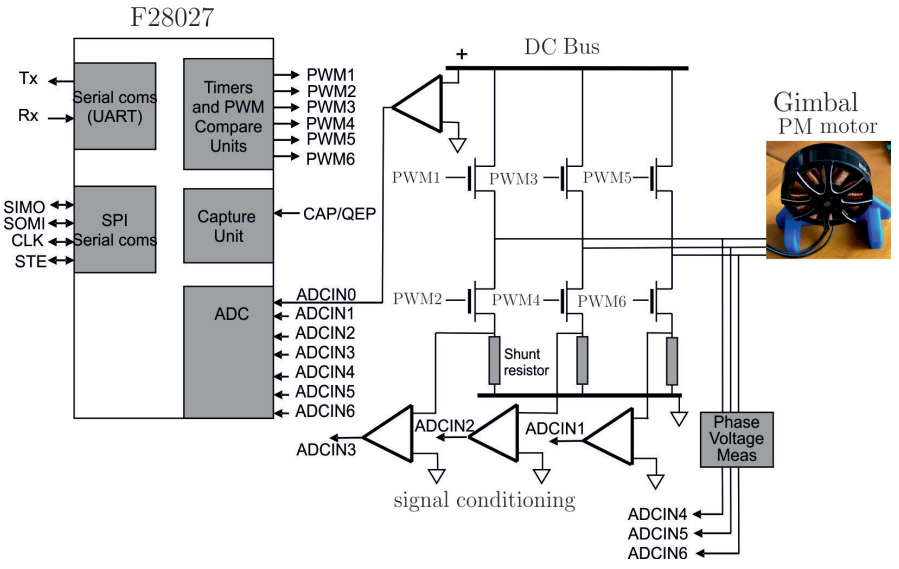


Fig. 2.2 Electrical drive setup

The hardware drive system setup, as shown for this gimbal drive example, reflects the type of setup needed to drive any three-phase permanent magnet or induction motor under sensorless operation. Developing the required control algorithms to be executed by the F28027 card (or any other MCU discussed in this GUIDE), is usually the most challenging aspect of drive development. When using the InstaSPIN-FOC or Altair AFT technology, all of the functionality needed to achieve sensorless control is provided. However it is still instructive, and at times desirable, to understand and execute the basic control functions, such as field-oriented current control and speed control. Hence the basic algorithms and principles, are outlined in the following subsections.

2.1.2 Use of a Speed Control loop

Speed control for any drive is of primary concern and a typical example, as given in figure 2.3, shows an ideal machine that is able to generate a machine torque T_{me}. The machine is connected to a load torque T_l and is assumed to have a lumped inertia of J. The relationship between machine/load torque and inertia is represented by equation 2.1.

$$T_{\mathrm{me}} - T_l = J\frac{d\omega_{\mathrm{me}}}{dt} \tag{2.1}$$

The mechanical shaft speed ω_{me} rad/s can also be expressed as $n_{\mathrm{me}} = \omega_{\mathrm{me}}\, {}^{30}/_{\pi}$ rpm for convenience.

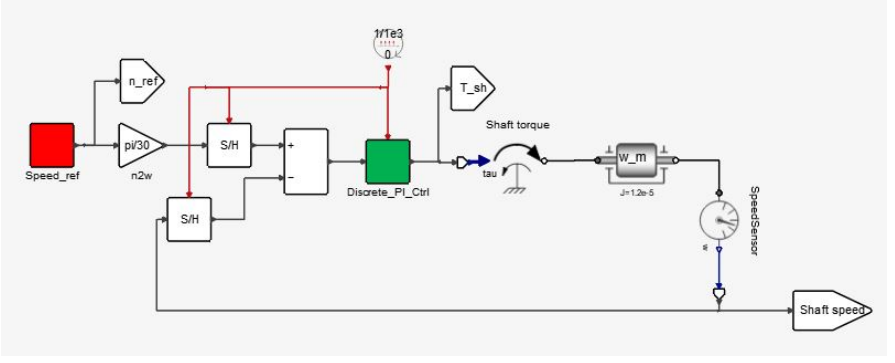


Fig. 2.3 ACTIVATE®based Speed Control loop example

In this example it is assumed that the ideal machine can deliver the required mechanical reference torque T^*_{me} generated by the 'PI' speed controller, hence the following condition applies

$$T_{\mathrm{me}} = T^*_{\mathrm{me}} \tag{2.2}$$

The speed controller, is typically a 'Proportional-Integral' (PI)[12] concept, which may be represented by the expression

$$T_{\mathrm{me}} = K_{\mathrm{p}}\Delta\omega_{\mathrm{me}} + K_{\mathrm{i}}\int \Delta\omega_{\mathrm{me}}dt \tag{2.3}$$

where K_{p} and K_{i} represent the proportional and integral gains of the speed controller respectively. The variable $\Delta\omega_{\mathrm{me}}$ is the mechanical speed error which is the difference of the user defined reference speed ω^*_{me} and measured shaft

speed (from the attached motor) ω_{me}. Suitable values for the proportional speed gain and integral speed gain can according to [13] be expressed as,

$$K_{\mathrm{p}} \approx J\,\omega_b \tag{2.4}$$

$$K_{\mathrm{i}} \approx J\,\frac{\omega_b^2}{4} \tag{2.5}$$

where ω_b is the user defined speed controller bandwidth, which is typically in the order of 100 rad/s. Note that the ratio between K_{p} and K_{i} is equal to $\frac{\omega_b}{4}$, which is a convenient factor to take into account when setting the integral speed gain, relative to the proportional value used. An ACTIVATE simulation, as given in figure 2.4, shows the speed response (measured shaft speed shown in 'red') and torque reference output of the speed loop.

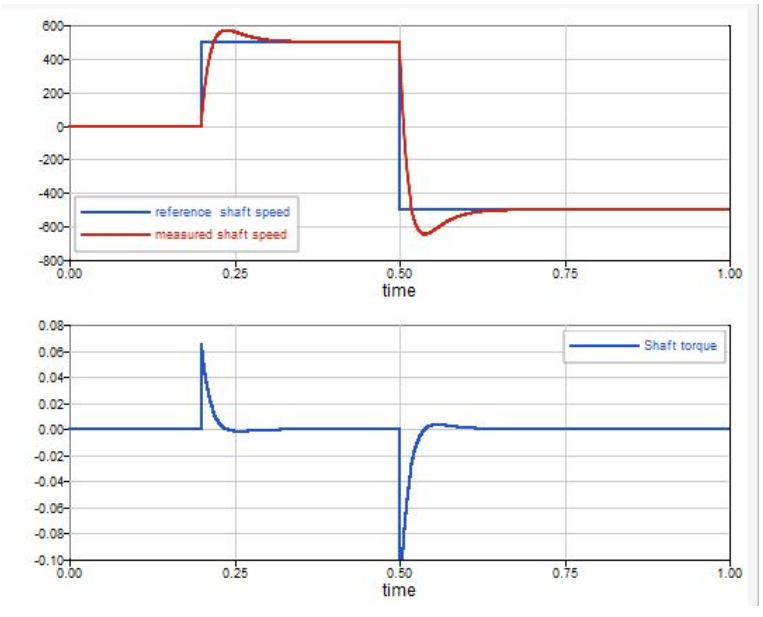


Fig. 2.4 Speed and torque time response

This example uses an inertia value of J =12 μkgm^2 (in the order of the value found for the PM Gimbal motor), which according to equation (2.4) corresponds to a proportional speed gain of K_{p} =1.2 mNm/(rad/s) and integral gain K_{i} =30 mNm/rad, where a user defined speed bandwidth of ω_b =100 rad/s is assumed. The controller in use is provided with an output limit capability,

which in this case is set to ±100 mNm. Note that in the example shown, the load torque T_l is assumed to be zero, i.e motor operating under no load and undergoing a speed step from $0 \rightarrow 500$ rpm followed by a speed reversal $500 \rightarrow -500$ rpm at $t \approx 0.5$ s. A discrete, fixed point implementation of the same example, with a sampling frequency of $f_s = 1\,kHz$ (which corresponds to a time period $T_s = {}^1/_{f_s}$) can also be undertaken in Embed, which requires scaling and discretization of expression (2.3) using the approach discussed in subsections 1.4.2 and 1.4.3. In terms of scaling it is helpful to introduce a full-scale torque value T_{fs} set to 0.150 Nm and full-scale frequency f_{fs} set to 400 Hz, given that speed is represented as a (mechanical) frequency in Hz. Subsequent mathematical handling gives

$$T^n_{\text{me}}(t_k) \cong \underbrace{\left(\frac{2\pi K_{\text{p}} f_{\text{fs}}}{T_{\text{fs}}}\right)}_{\text{Gain Kp}} \Delta f^n_{\text{me}} + \underbrace{\left(\frac{2\pi K_{\text{i}} T_{\text{s}} f_{\text{fs}}}{T_{\text{fs}}}\right)}_{\text{Gain K}_{\text{i}}} \Delta f^n_{\text{me}} + T^n_{\text{i}}(t_{k-1}) \qquad (2.6)$$

where $\Delta f^n_{\text{me}} = \frac{1}{60\, f_{\text{fs}}}\,(n^*_{\text{me}} - n_{\text{me}})$ is the per unit speed error. The last two terms of equation (2.6) represent the integrator torque output component $T^n_{\text{i}}(t_k)$, whilst $T^n_{\text{i}}(t_{k-1})$ is the 'before last' integrator contribution generated using a '`udely` module' as shown in figure 2.5.

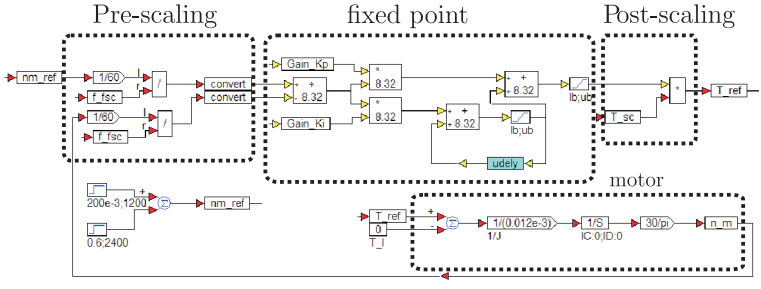


Fig. 2.5 Embed Speed control loop example

Observation of figure 2.5 shows the presence of the fixed point 'PI' controller, which is implemented in IQ24 in this case given that the numerical Gain Kp value, as calculated using equation (2.6) exceeds the +1.9999 value available in IQ30, given the full-scale values used. Two limiter modules with max/min range: `ub`,`lb` are added to the fixed point controller to avoid 'integrator windup' [13] and limit the per unit output torque T^n_{me} to $\pm\left(\frac{0.1}{T_{\text{fs}}}\right)$, in accordance with the values used for the ACTIVATE example.

A pre-scaling and post-scaling unit are again used to interface between fixed point and non-scaled variables. The motor is modelled in the floating point, continuous time domain. Results achieved with this simulation, using the same

reference speed versus time sequence as used for the ACTIVATE example, are, as may be expected, identical.

2.1.3 Overview of DC and AC machines

Central to the ability to achieve accurate speed control is the requirement to satisfy equation (2.2), which states that the machine must be able to deliver the torque specified by the speed controller. Achieving torque control, essentially implies controlling the orientation and magnitude of a current distribution (in either stator or rotor) relative to a given rotating magnetic field [13] of the machines considered in the GUIDE.

In brushed DC machines, a magnetic field is present in the stator and generated by either a field winding or permanent magnets (as discussed in the GUIDE). A brush/commutator assembly is used to achieve a stationary current distribution with a rotating rotor (armature). This approach basically ensures that the orientation of the current vector is orthogonal to the magnetic flux vector, which is required for optimum (highest torque for the lowest current) torque production [12].
In the sequel of this section attention is focussed on the torque production mechanism of the AC motors shown in figure 2.6.

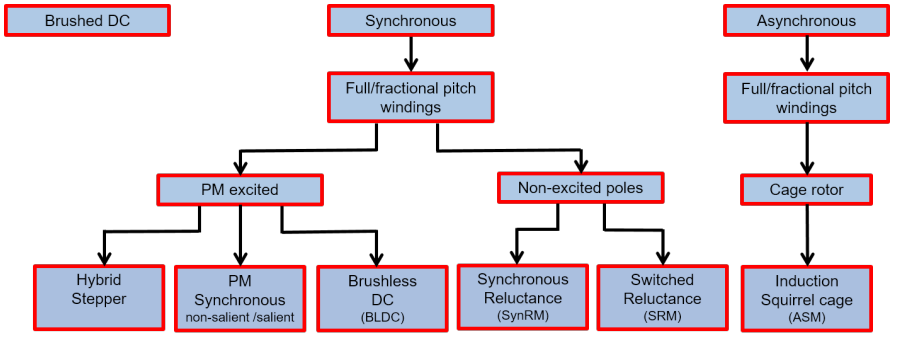
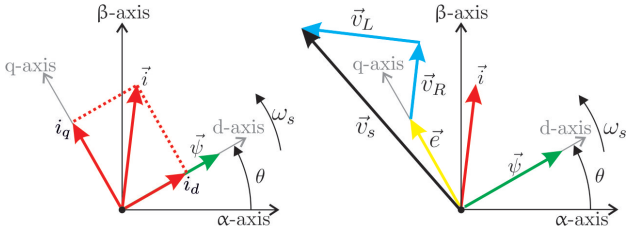


Fig. 2.6 Overview of DC& AC motors used in the GUIDE

Indicated are two categories named 'Synchronous' and 'Asynchronous' respectively, which defines if an applied rotating current vector $\vec{i}$ needs to be stationary relative to the rotating rotor in order to produce a constant torque. For synchronous motors this is indeed the case, while in induction machines the current vector must typically move at a 'slip-speed' relative to the rotor. In the ensuing sub chapters a overview of the motor concepts will be briefly given in terms of topology and torque production mechanism. In addition, so called 'terminal vector plots' are used to show the flux, current and terminal

voltage vectors, which is important for understanding field-weakening, sensorless control and general operational aspects of electrical drives. Included in figure 2.6 is the switched reluctance machine, which is normally used with an asymmetric converter[13]. However, in this GUIDE it is used with a three-phase converter, which is a non-standard approach akin with behaviour as a synchronous drive. This approach is used to allow sensorless FOC of a four phase SRM machine with a three-phase converter.

The key to understanding torque production of the AC machines shown in figure 2.6 is to consider machine models with two magnetic poles, i.e a one pole pair $p = 1$. Of specific interest is the examination of the relationship between the flux and current vectors [12] as shown in figure 2.7(a). Shown in



(a) Flux-current vector plot (b) Terminal vector plot

Fig. 2.7 Flux-current and Terminal vector plots

figure 2.7(a) are a vector $\vec{\psi}$ which represent the magnetic field of the two-pole machine and this is aligned with the d-axis of a 'd,q' orthogonal system. i.e the angle between the two axis is $\frac{\pi}{2}$. The term 'field-oriented control' (FOC) focusses on the orientation of the current vector $\vec{i}$ generated by the controller relative to said flux vector ψ.
The electromagnetic torque T_e, as defined by equation (2.7a) [12],[14] shows that the projection of the current vector onto the 'q-axis' times the flux amplitude ψ defines its value.

$$T_e = \frac{3}{2}\Im\left\{\vec{\psi}^*\,\vec{i}\right\} = \frac{3}{2}\left(i_q\,\psi\right) \tag{2.7a}$$

$$\omega_s = \frac{d\theta}{dt} \tag{2.7b}$$

The factor 3/2 is introduced in equation (2.7a), because we are using throughout this GUIDE, so called 'amplitude-invariant', vectors [12]. The rotational 'electrical ' speed ω_s of the flux vector (and the d,q coordinate system) is

determined by the differential of the angle θ (see equation (2.7b)). This angle represents the orientation of the 'd-axis' relative to the stationary α, β reference frame, which is tied the stator of the machine. For implementation of field-oriented control, the 'd-axis' must be aligned with the flux vector $\vec{\psi}$.

To gain a better understanding of machine operation it is prudent to take into account the fact that the terminal back-EMF of the machine is linked to the applied terminal voltage by a series network consisting of an inductance L and stator resistance R_s as shown in figure (2.8). This relationship is represented

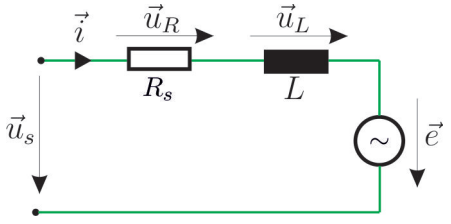


Fig. 2.8 Typical machine model

by equation (2.8), and also by a so called 'terminal vector plot' as shown in figure 2.7(b).

$$\vec{u}_s = \vec{u}_R + \vec{u}_L + \vec{e} \tag{2.8a}$$
$$\vec{e} = j\omega_s \vec{\psi} \tag{2.8b}$$
$$\vec{u}_R = \vec{i}\, R_s \tag{2.8c}$$
$$\vec{u}_L = j\omega_s L \vec{i} \tag{2.8d}$$

The vectors $\vec{u}_R$ and $\vec{e}$ represent the voltage across the stator resistance and the EMF, which is a function of the electrical frequency ω_s and flux ψ. In fact for many sensorless control algorithms defining the orientation of the EMF vector is a critical step to determining the location of the flux axis and therefore the angle θ.

In due course the topic of 'field-weakening' will be introduced, which makes use of the voltage across the inductance L that is defined as u_L. However, by introducing the short-circuit current i_{sc} as defined by equation (2.9) a more revealing expression is found (see equation (2.10).

$$i_{\mathrm{sc}} = \frac{\psi}{L} \tag{2.9}$$

Expression (2.10) shows that the amplitude of the voltage vector across the inductance is equal to the amplitude of the EMF vector TIMES a factor defined by the ratio of the current $|\vec{i}|$ and shortcircuit current i_{sc}.

$$\vec{u}_L = j \left(\frac{\vec{i}}{i_{\mathrm{sc}}} \right) |\vec{e}| \tag{2.10}$$

Consequently, if the short circuit current of a machine is high in comparison to the current amplitude, then the inductance voltage amplitude will be low RELATIVE to the EMF amplitude. This is a critical observation when using field-weakening which relies on the presence of a representative u_L value (when compared to the EMF amplitude).

Finally, it is noted that many machines in this GUIDE will be multi-pole, hence $p > 1$ (where p is the number of magnetic pole pairs), in which case the mechanical shaft speed ω_{me} and shaft torque T_{me} will be as indicated in equation (2.11).

$$T_{\mathrm{me}} = p\, T_{\mathrm{e}} \tag{2.11a}$$

$$\omega_{\mathrm{me}} = \frac{\omega_{\mathrm{s}}}{p} \tag{2.11b}$$

$$\text{angle between (+d, +q) axis} = \frac{\pi}{2\,p} \tag{2.11c}$$

Note also that in multi-pole machines the actual angle between the d-q axis will not be $\pi/2$, hence for control purposes the machine is viewed (internally) as two pole, in order to work with orthogonal coordinate systems.

Another important merit indicator is the so called 'EMF threshold speed' n_{e}, which is the shaft speed (rpm) where the EMF amplitude e becomes equal

to the voltage drop $u_R = i^{\text{Rated}} R_s$. This can with the aid of equations (2.8), (2.9) be calculated, which leads to expression (2.12).

$$n_{\text{e}} = \left(i^{\text{Rated}}/i_{\text{sc}}\right)\, n_{\text{sc}} \tag{2.12a}$$

$$n_{\text{sc}} = \frac{30\, R_s}{\pi\, p\, L} \tag{2.12b}$$

where p is the machine pole pair number (see equation 2.11) and i^{Rated} the rated stator current amplitude. Also included in expression (2.12) is a variable n_{sc}, which is the shaft speed needed to generate an EMF that is equal to $e = i_{\text{sc}}\, R_s$, where i_{sc} is the short-circuit current (see equation (2.9)).
The EMF threshold speed n_{e} is a good indicator when dealing with control of sensorless (encoderless) drives, as will be discussed at later stage in the GUIDE. Preferably, a low EMF threshold speed is desirable as this means that the impact of the stator resistance is limited for the flux estimator.

2.1.3.1 Brushed DC

The brushed DC machine shown in figure 2.9 contains a rotor (referred to as the 'armature') with a set of windings that is connected via a 'brush-commutator' assembly to the two terminals where the brushes are located. The purpose of this assembly is to ensure that current distribution in the armature remains STATIONARY whilst the armature is rotating. This current distribution is represented by a current vector in the stationary coordinate system, which is along the 'q-axis'. The flux-linkage ψ_{PM} due to the permanent magnet (or a field-winding) corresponds to a flux vector oriented along the 'd-axis' as may be observed from figure 2.9. The orientation of the flux and current vector is orthogonal, which is the optimum setting to produce the highest shaft torque for the lowest armature current. This 'field-orientation' is precisely what is also present in AC machines, as will be discussed at a later stage.
A symbolic representation of the brushed machine, as given in figure 2.10 corresponds with the equation set defined by 2.13.

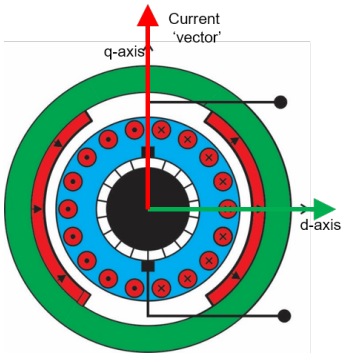


Fig. 2.9 Brushed DC with PM stator field

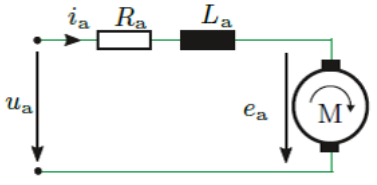


Fig. 2.10 Symbolic model Brushed DC machine

$$u_a = i_a R_a + L_a \frac{di_a}{dt} + e_a \tag{2.13a}$$

$$e_a = \omega_m \psi_{\mathrm{PM}} \tag{2.13b}$$

$$T_e = \psi_{\mathrm{PM}} i_a \tag{2.13c}$$

$$T_e - T_l = J \frac{d\omega_m}{dt} \tag{2.13d}$$

This equation set is also used for the construction of a generic model, as used in the SIM, PIL and COSIM toolchain elements and leads to the representation shown in figure 2.11.

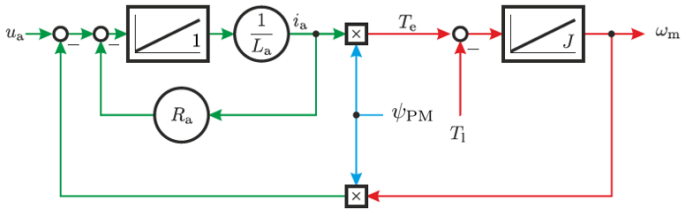
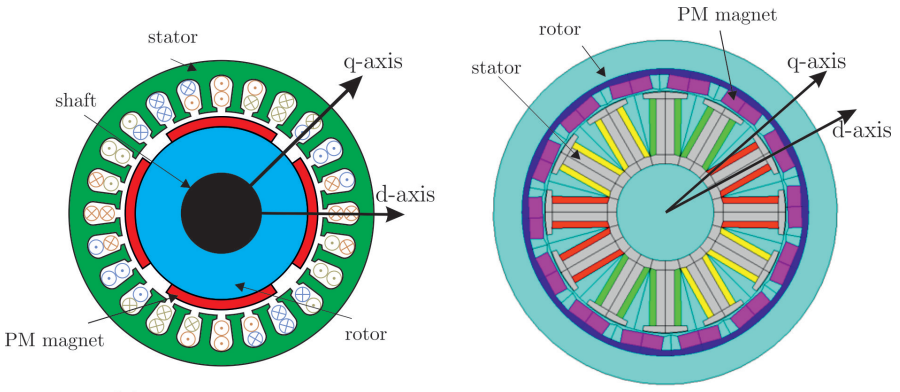


Fig. 2.11 Generic brushed DC model [13]

2.1.3.2 Non-salient PM: PMsyn and BLDC

The term 'non-salient' refers to the inductance that is linked to the 'd' and 'q' axis. If these are found to be equal then the machine is deemed to be non-salient. An indication of inductance differences between the 'd' and 'q' axis can be observed by considering the path an externally applied flux (ignoring the presence/effects of the magnets) and looking at the magnetic resistance encountered along either axis. In this experiment take into account that the relative permeability of permanent magnets is virtually identical to air. For example the PM machine shown in figure 2.12(a) displays the '+d', '+q" axis (one set of two in this case). When moving along either axis, the magnetic resistance encountered is the same, hence the machine is deemed to be non-salient. A similar mind experiment can also be done for the adjacent motor shown in figure 2.12(b), which is an 'outrunner' type because the rotor is on the outside of the motor. This a seven pole pair $p = 7$ (14 magnets) and the angle between the 'd,q' axis shown is equal to 12.8 degrees, as calculated using equation (2.11c).

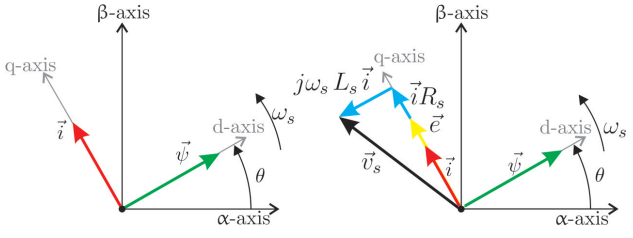
Both machines are three-phase, but they are also different (apart from the rotor being on the outside or inside of the stator) in terms of the winding configuration used. This phenomenon can be observed by looking at the EMF versus time plot with an oscilloscope and connecting two phase windings (which is a no load line-to line measurement). The machine in figure 2.12(a) is known as a PM synchronous machine because a distributed full pitch winding is used, which gives a quasi-sinusoidal EMF-time plot. PM machines with fractional concentrated windings [27] have a quasi-trapezoidal EMF, time plot and the corresponding EMF vector locus will be quasi-hexagonal, whereas PM synchronous (PMsyn) machines exhibit a circular EMF locus. Note also that the BLDC machines typically have a number of magnets that is NOT equal to the number of stator poles (see figure 2.12(b), which shows 14 magnets and 12 stator poles) to reduce torque cogging effects. For permanent magnet machines (BLDC or PMsyn), orientation of the current vector relative to the (PM generated) flux vector (see figure (2.13(a)) must be carried out by a control algorithm using either an encoder (so called sensored operation) or sensorless, as for example carried out by TI InstaSPIN®or AFT technology.



(a) 4 pole PMSyn (b) 14 pole Outrunner BLDC

Fig. 2.12 Non-salient PMsyn and BLDC examples

The challenge to be met is to generate a current vector which remains stationary relative to the magnetic flux vector for all speeds, using a three-phase (or two-phase) winding. Hence the need to control the current in each of the three phases with an attached power electronic converter. Under normal



(a) Flux-current vector plot (b) Terminal vector plot

Fig. 2.13 Flux-current and Terminal vector plots: PMSyn and BLDC

conditions the current vector is chosen to be perpendicular to the flux vector. This provides the best torque/current ratio, referred to as 'Max Torque Per Ampere' (MTPA) operation. The torque is thus equal to

$$T_e = \frac{3}{2}\,\psi\, i_q \tag{2.14}$$

where i_{q} is the quadrature axis component of the current vector $\vec{i}$, hence in this case $\vec{i} = j i_q$. The terminal vector plot (see see figure (2.13(b)), shows the use of the stator inductance L_s which is deemed equal for both d,q axis. In some cases, particularly at higher operating speeds, so-called 'field weakening' is used (which depends on the value of the short-circuit current relative to the rated machine current), which implies the presence of a direct axis current

component i_d that is purposely chosen to manipulate the voltage $\vec{u}_L$ across the inductance.

Note that BLDC machines are typically excited with a three-phase converter that in effect utilizes the quasi- hexagonal flux-linkage locus by six-stepping the current vector using Hall effect position sensors, as will be discussed at a later stage (see Hall sensored BLDC example).

The IRTF based equation set for the PMsyn [13] is as follows:

$$\vec{u}_s = R_s \vec{i}_s + \frac{d\vec{\psi}_s}{dt} \tag{2.15a}$$
$$\vec{\psi}_s^{xy} = \vec{\psi}^{xy} + L_s \vec{i}_s^{xy} \tag{2.15b}$$
$$\theta = p\,\theta_m \tag{2.15c}$$
$$T_m = p\,T_e \tag{2.15d}$$
$$T_m - T_l = J\,\frac{d\omega_m}{dt} \tag{2.15e}$$
$$\omega_m = \frac{d\theta_m}{dt} \tag{2.15f}$$

where $\vec{\psi}_s$ and $\vec{\psi}$ represent the stator and rotor flux respectively in stationary coordinates. The superscript xy refers to a vector in a synchronous reference frame that is tied to the rotor of the machine.
Shown in figure 2.14 is the corresponding generic model of the PMsyn machine. For BLDC models the generic model is extended by adding a six harmonic component to the vector $\vec{\psi}^{xy}$, which is a function of the electrical angle θ. The IRTF models for non-salient PM machines have an option, which allows the user to select either a PMsyn or BLDC model.

Central to the generic model is the 'Ideal Rotating TransFormer' module, which is defined by the following equation set:

$$\vec{\psi}_s^{xy} = \vec{\psi}_s\, e^{-j\theta} \tag{2.16a}$$
$$\vec{i}_s = \vec{i}_s^{xy}\, e^{j\theta} \tag{2.16b}$$
$$T_e = \frac{3}{2}\Im\left(\vec{\psi}_s^{*}\,\vec{i}_s\right) \tag{2.16c}$$

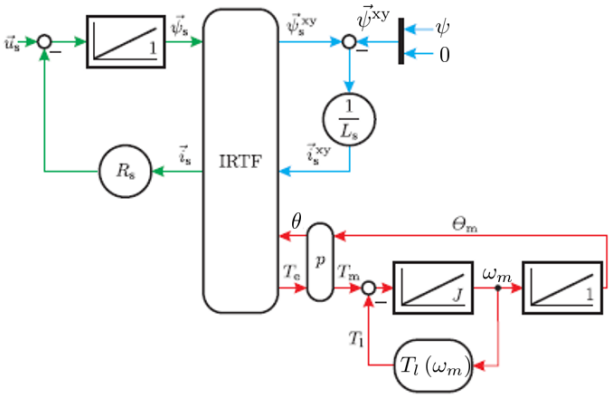
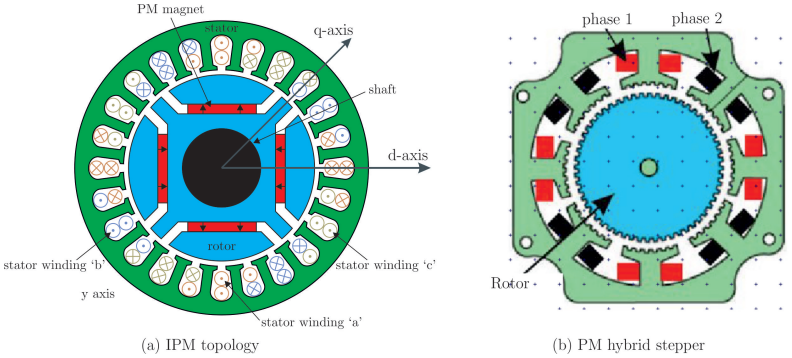


Fig. 2.14 IRTF based Generic model: PMSyn [13]

Where it is noted that equation 2.16c is in fact equation 2.14, but written in a different way.

2.1.3.3 Salient PM: IPM and hybrid stepper

The term 'salient' refers to the presence of unequal inductance values between the 'd' and 'q' axis as may be observed from the two pole pair $p = 2$ example, shown in figure (2.15)(a).



(a) IPM topology (b) PM hybrid stepper

Fig. 2.15 IPM topology and two-phase PM hybrid stepper

When moving along, for example, the 'd' axis in a mind experiment, the observed magnetic reluctance will be higher when compared to the same scenario for the 'q axis' given the fact that the latter path encounters LESS air. Consequently L_d will be lower then L_q, given that lower reluctance value corresponds to a higher inductance [13]. Hence the so called 'incremental

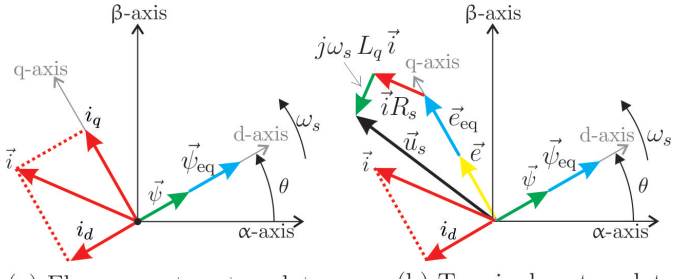
inductance' defined by equation (2.17) will be NEGATIVE for the type of machine shown in figure (2.15)(a).

$$\Delta L = (L_d - L_q) \tag{2.17}$$

The term 'interior permanent magnet' (IPM) refers to a PM topology, where the magnets are located inside the rotor, i.e not located in the airgap, as is the case for a PMsyn machines. Pending the value of the incremental inductance two types of IPM's are present namely: 'flux-enhancement', $\Delta L > 0$ and 'flux-weakening', $\Delta L < 0$. A special case of 'IPM' is the hybrid stepper shown in figure (2.15)(b), this is not an IPM in terms of topology but its behaviour from a control perspective resembles a two -phase flux-enhancement type IPM, as will become apparent at a later stage. Most IPM's typically utilize a three-phase distributed winding, like the example shown in figure (2.15)(a).

When considering the torque production of an IPM it is again convenient to look at the flux-current plot, see figure (2.16)(a). This type of machine has the ability to generate an 'equivalent flux' component ψ_{eq} [13] as indicated in equation 2.18b. This additional (to the PM flux ψ component) is due to the presence of salient inductance, which corresponds to a non zero incremental inductance. If for example $\Delta L < 0$ (as is assumed here) a NEGATIVE current component i_d is required to generate an equivalent flux component that enhances the PM flux value, as may be observed from figure (2.16)(a).

The torque generated by an IPM is given by the projection of the current vector $\vec{i}$ on the 'q-axis' (known as i_q) times the sum of the two flux components as indicated by equation (2.18a), which shows two torque components defined as the 'electrodynamic torque' T_{ed} and 'equivalent torque' T_{eq} respectively. The control challenge for IPM drives is to identify the optimum



(a) Flux-current vector plot (b) Terminal vector plot

Fig. 2.16 Flux-current and Terminal vector plots: IPM, case $\Delta L < 0$

i_d, i_q values that must be used to achieve the best 'maximum torque per ampere'(MTPA), which is a function of the assigned torque reference, PM flux level and incremental inductance value [13].

$$T_{\mathrm{e}} = \frac{3}{2}\left(\underbrace{i_q\,\psi}_{T_{\mathrm{ed}}} + \underbrace{i_q\,\psi_{\mathrm{eq}}}_{T_{\mathrm{eq}}}\right) \tag{2.18a}$$

$$\psi_{\mathrm{eq}} = i_d\,\Delta L \tag{2.18b}$$

The terminal vector plot shown in figure (2.16)(b) is based on the equivalent circuit model given by figure (2.8), where for IPM machines the inductance L is usually chosen to be quadrature inductance L_q. If this approach is taken, the resultant EMF will consist of TWO components namely due to the PM flux $\vec{e} = j\omega_s\,\vec{\psi}$ AND the equivalent flux $\vec{e}_{\mathrm{eq}} = j\omega_s\,\vec{\psi}_{\mathrm{eq}}$, where the latter is a function of the current i_d (see equation (2.18b). It is emphasized that this formulation of the terminal plot shown relies on the use of a single inductance representation (L_q in this case) in the equivalent circuit. Under these circumstances the EMF that will estimated is equal to $\vec{e}_{\mathrm{eq}} + \vec{e}$. Correspondingly, the flux estimate will then be equal to $\psi_{PM} + \psi_{eq}$.

A generic IRTF model of an IPM is based on the PMsyn model [13] (see figure 2.14). The important change is the inclusion of two salient inductance variables on the 'xy' (also referred to as'dq' coordinates) side of the IRTF. Hence, calculation of the vector $\vec{i}_s^{xy} = \left(\frac{(\psi_{sx}-\psi)}{L_d},\ \frac{\psi_{sy}}{L_q}\right)$, as shown in figure 2.17.

Magnetic saturation is often an issue for this type of machine, in which case the linear model needs to be adapted [29] by making use of two dimensional functions $i_{sx}\left(\psi_{sx},\,\psi_{sy}\right), i_{sy}\left(\psi_{sx},\,\psi_{sy}\right)$ as shown in figure 2.18. This approach requires access to the magnetization curves $\psi_{sx}\left(i_{sx},\,i_{sy}\right), \psi_{sy}\left(i_{sx},\,i_{sy}\right)$, which are generated using 'FluxMotor' [19]. Subsequently, these characteristics need to be inverted [29] for the IRTF model indicated in figure 2.18.

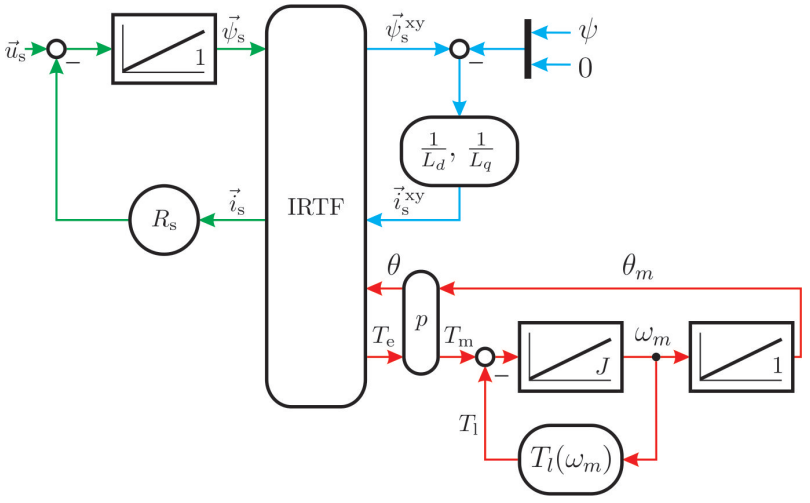


Fig. 2.17 Linear IRTF based IPM model [13]

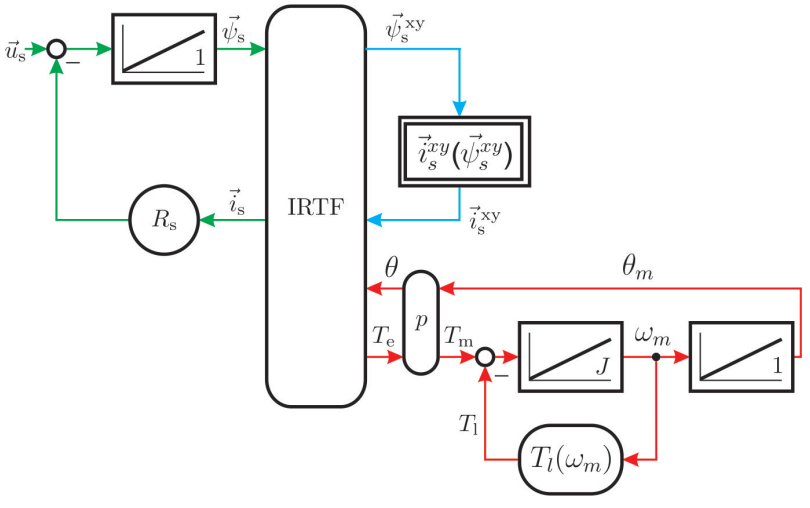
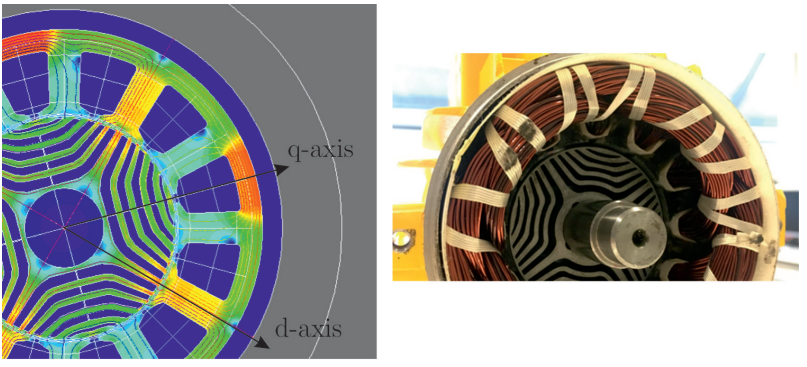


Fig. 2.18 IRTF based IPM model, with magnetic saturation [29]

2.1.3.4 Synchronous Reluctance Motor (SynRM)

The synchronous reluctance machine, as shown in figure (2.19)(a), has either a distributed or fractional pitch three-phase winding on the stator, similar to PMSyn machines. The key difference is that the rotor design must be configured to optimize the ratio between direct and quadrature inductance values. Observation of figure (2.19) shows that the magnetic reluctance along the 'd-axis' is notably lower then the 'q-axis', which is aimed at optimizing the incremental inductance $\Delta L = L_d - Lq$ value. Also shown in figure (2.19)

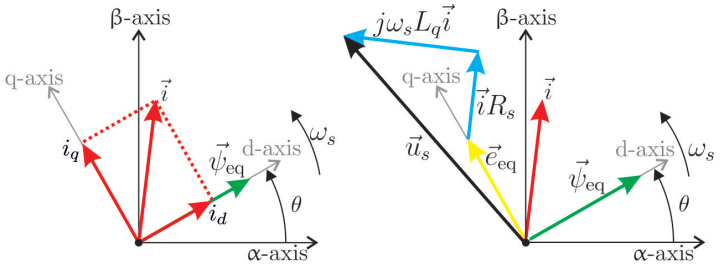
is a practical implementation [24] of the model given in figure (2.19)(a), see figure (2.19)(b).



(a) SynRM topology (b) SynRM example

Fig. 2.19 Synchronous Reluctance Machine (SynRM) [24]

In terms of torque production, the equivalent flux term ψ_{eq} relies on the use of a direct current component, as may be observed from figure (2.20)(a). The quadrature current current component required to generate the torque is indicated in expression (2.19).



(a) Flux-current vector plot (b) Terminal vector plot

Fig. 2.20 Flux-current and Terminal vector plots: SynRM

Control as such is very similar to the induction machine, to be discussed shortly.

$$T_{\text{e}} = \frac{3}{2}\, i_q\, \psi_{\text{eq}} \tag{2.19a}$$

$$\psi_{\text{eq}} = i_d\, \Delta L \tag{2.19b}$$

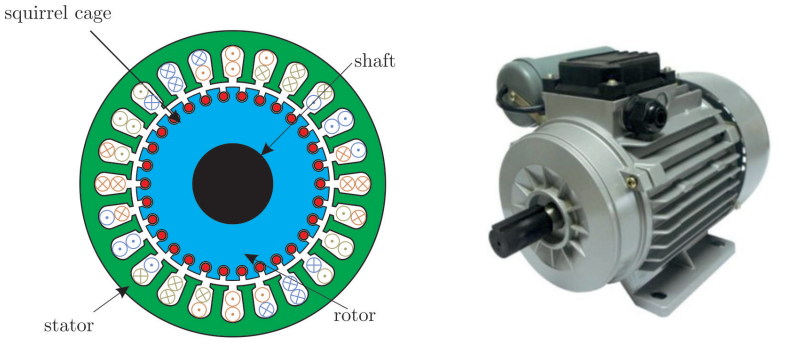
However, it is important to note that the incremental inductance value ΔL is prone to saturation effects, as is also the case for IPM machines.

The terminal vector plot shown in figure (2.20)(b) is , shows that the EMF vector $\vec{e}_{\text{eq}} = j\,\omega_s \vec{\psi}_{\text{eq}}$ is linked to the terminal voltage vector by the sum of the voltage components across the quadrature inductance and stator resistance respectively. For this machine an equivalent short-circuit current can also be defined as $i_{\text{sc}}^{\text{eq}} = \frac{\psi_{\text{eq}}}{Lq}$. Typically the short-circuit current is low relative to the rated stator current hence the voltage component $\vec{u}_L$ can be substantial (for low ΔL values), which in turn leads to a lower power factor. However, the synchronous reluctance machine is becoming increasingly popular due to the absence of (costly) permanent magnets. The IRTF model for this machine is very similar to the IPM models shown in figures 2.17, 2.18 with the important difference that the variable ψ representing the permanent magnet flux magnitude, must now be set to zero, as no magnets are present in this type of machine.

2.1.3.5 Induction machines

The induction motor (IM), otherwise known as an 'asynchronous' machine has a stator with a three-phase distributed winding as shown in figure (2.21)(a) and a 'squirrel cage' rotor [12]. The number of pole-pairs of this machine is defined by the winding configuration used. A single-phase induction machine (see figure (2.21)(b)), is commonly connected to a single-phase mains supply and has a capacitor (typically located in the cylindrical housing shown on the top) which is used to generate a phase current difference between the two stator windings. In this GUIDE such a machine will be connected to a three-phase converter WITHOUT the capacitor, in which case the machine can be seen as a two-phase IM motor.

Torque production in an induction motor is slightly more complicated when compared to machines discussed above. The primary reason for this is that the controller must generate a stator current distribution that will produce a magnetic field (given that there are no magnets or field winding) and generate a constant torque. Given this dual task it is convenient to represent and discuss both components of the current distribution separately.
The two figures given in figure 2.22 show the stator current distributions in question for a two pole (one pole pair) machine. In reality both stator current distributions are superimposed to form one single current distribution as will be discussed at a later stage. The stator current distribution in the 'left' figure is responsible for the generation of the magnetic flux and correspondingly the



(a) Induction machine topology (b) Single-phase Induction machine

Fig. 2.21 Induction machine topology and single-phase induction example

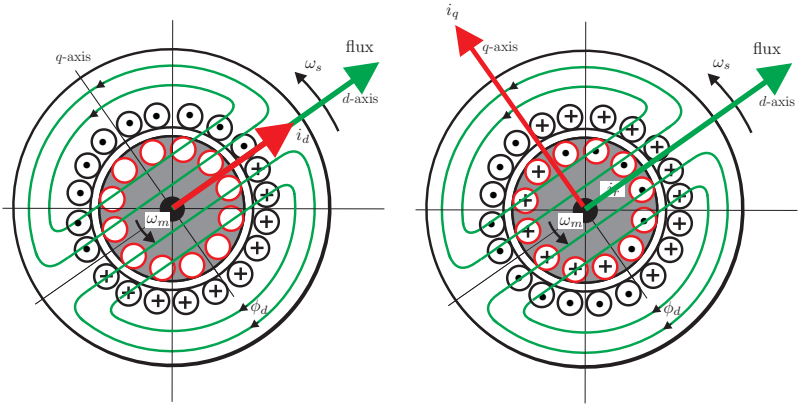
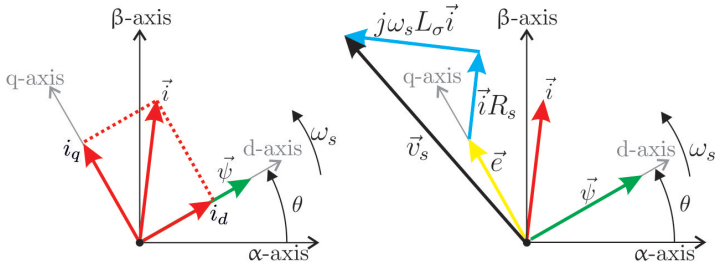


Fig. 2.22 Torque and flux generating mechanisms in a IM (asynchronous) machine

current vector and flux vector are both oriented along the d-axis. Torque production is achieved by orientating a second current distribution perpendicular to the flux, hence along the q-axis as shown in the 'right' figure. However, in an induction machine this q-axis current distribution must be mirrored in the rotor because there is no resultant flux in the 'q' direction. To maintain the current distribution in the rotor a voltage must be induced in said rotor by the magnetic flux. This means that the magnetic field (and therefore the flux vector) must rotate relative to the stator. Hence the following relationship must hold to achieve field oriented control

$$\omega_\mathrm{s} - \omega_\mathrm{m} = \frac{i_q R_R}{\psi} \tag{2.20}$$

where R_{R} is the rotor resistance of the squirrel cage rotor (to be discussed further in the next section). The resultant stator current distribution in the machine is now formed by the superposition of the two stator current distributions represented by the vector $\vec{i}$, which in turn has a direct and quadrature current components, as shown in figure (2.23)(a).



(a) Flux-current vector plot (b) Terminal vector plot

Fig. 2.23 Flux-current and Terminal vector plots: IM

The torque of the machine is given by product of the rotor flux and quadrature current, which leads to

$$T_{\mathrm{e}} = \frac{3}{2} i_q \psi \tag{2.21a}$$

$$\psi = i_d L_{\mathrm{M}} \tag{2.21b}$$

For this type of machine the flux magnitude is controlled by the direct axis current i_d and the magnetizing inductance L_M. Often the relationship between the rotor flux and magnetizing current is not linear due to saturation effects. The terminal vector plot shown in figure (2.23)(b) indicates that the EMF vector $\vec{e} = j\,\omega_s\,\vec{\psi}$ is linked to the terminal voltage vector $\vec{u}_s$, by the sum of the voltage vector components tied to the leakage inductance L_σ and stator resistance R_s respectively (see figure 2.23(b)).

To achieve a torque step in an induction machine means increasing the current i_q value and simultaneously increasing the speed ω_s of the reference frame to which the current and flux vectors are tied. Alternatively the flux could also be increased, but this cannot be be done instantaneously as this event is tied to a time constant set by the magnetizing inductance divided by the rotor resistance. In the latter case the stator frequency must, according to equation (2.20) be reduced, given that the q-axis current remains con-

stant. Reducing the flux (known as field weakening) is often undertaken at high operating speeds given the inherent DC bus voltage limitations of a drive.

The model structure according to figure 2.8 can also be applied to an induction machine. However, in this case it is helpful to consider the steady-state single phase four parameter model [13] given in figure 2.24. Shown in this

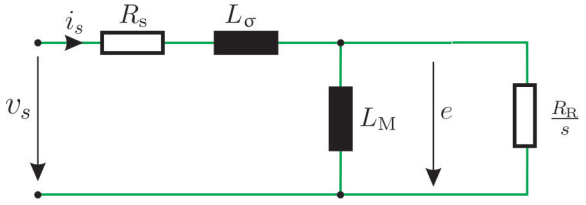


Fig. 2.24 4 parameter IM motor model (one phase representation)

figure are magnetizing inductance Ł$_M$ and combined leakage inductance Ł$_\sigma$, associated with the stator and (squirrel cage) rotor. In addition, the model shows the resistance R_s, R_R linked to the stator windings and squirrel cage rotor respectively.
In this type of model the effective rotor resistance (for given shaft speed n_m rpm) is determined by $^{R_R}/_s$, where s represents the 'slip' defined as

$$s = \frac{n_\mathrm{s} - n_\mathrm{m}}{n_\mathrm{s}} \tag{2.22}$$

where n_s rpm, represents the synchronous shaft speed. The latter is tied to the electrical supply frequency according to $n_\mathrm{s} = {}^{60\,f_s}/_p$, where f_s, p are the electrical frequency (Hz) and pole pair number respectively. For motor simulation purposes, an 'Ideal Rotating Transformer' (IRTF) space vector model (as introduced earlier for synchronous machines), is also used for representing a 4 parameter squirrel case induction machine[13].

The model, as shown in figure 2.25 is defined by equation set (2.23).

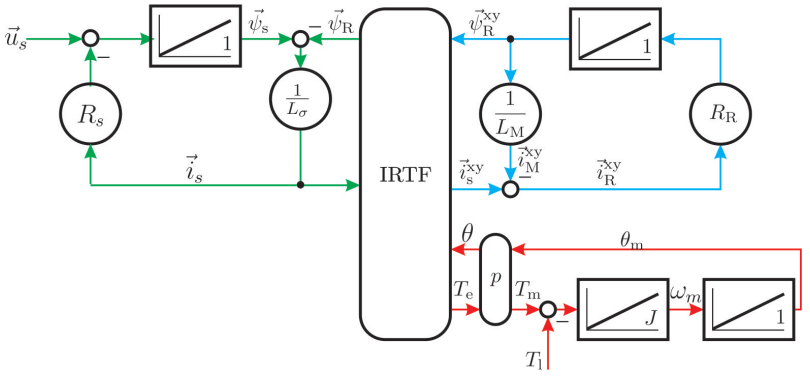


Fig. 2.25 Generic IRTF based Induction machine model: three phase

$$\vec{u}_s = R_s \vec{i}_s + \frac{d\vec{\psi}_s}{dt} \tag{2.23a}$$

$$\vec{\psi}_s = \vec{\psi}_R + L_\sigma \vec{i}_s \tag{2.23b}$$

$$\vec{i}_R^{xy} R_R = \frac{d\vec{\psi}_R^{xy}}{dt} \tag{2.23c}$$

$$\vec{i}_R^{xy} = \vec{i}_s^{xy} - \frac{\vec{\psi}_R^{xy}}{L_M} \tag{2.23d}$$

$$\theta = p\,\theta_m \tag{2.23e}$$

$$T_m = p\,T_e \tag{2.23f}$$

$$T_m - T_l = J\,\frac{d\omega_m}{dt} \tag{2.23g}$$

$$\omega_m = \frac{d\theta_m}{dt} \tag{2.23h}$$

A 'single' phase induction machine, is named according to the power supply used, as shown in figure 2.26. The machine is, however, configured with two distributed windings referred to as the 'run' and 'auxiliary' winding. Typically, a centrifugal switch in series with a capacitor is used to connect the auxiliary winding at start-up, to ensure the desired direction of rotation.

In this GUIDE drive operation of a single phase induction machine is carried out with the aid of a three-phase converter, with a revised wiring diagram as shown in figure 2.27. It is important to carefully identify the terminals used for two-phase control. In this case terminal 'C' is the 'common' phase connection for both windings.

The generic IRTF model of the single phase machine is similar to the three-phase example shown above (see figure 2.25). Note that the factor $(\frac{3}{2})$ must

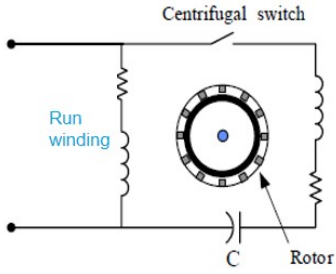


Fig. 2.26 Typical single phase induction machine: wiring diagram

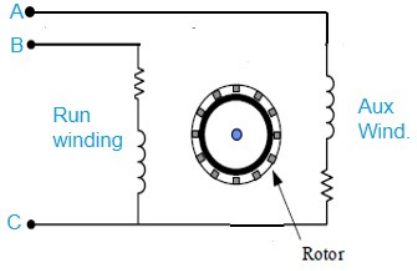


Fig. 2.27 Single phase induction machine for inverter use: revised wiring diagram

be removed from equation (2.21a) as this machine has two phase windings. Furthermore, the winding ratio needs to be accommodated in the single phase model by the inclusion of an 'Ideal Transformer' (ITF), as shown in figure 2.28. The ITF module, is defined by equation set (2.24).

$$\psi_{s\alpha} = \frac{1}{k}\psi_{aux} \tag{2.24a}$$

$$i_{s\alpha} = k\, i_{aux} \tag{2.24b}$$

$$\psi_{s\beta} = \psi_{run} \tag{2.24c}$$

$$i_{s\beta} = i_{run} \tag{2.24d}$$

where k is the ratio between the auxiliary winding turns and run winding turns.Typically this value is in the range $1 - 1.5$. Furthermore, the winding resistance of both windings is different and must therefore also be accommodated in the single phase IM model, as shown in figure 2.28.

Central to achieving precise torque control in machines discussed above is thus tied to the ability to correctly orientate and control the stator current

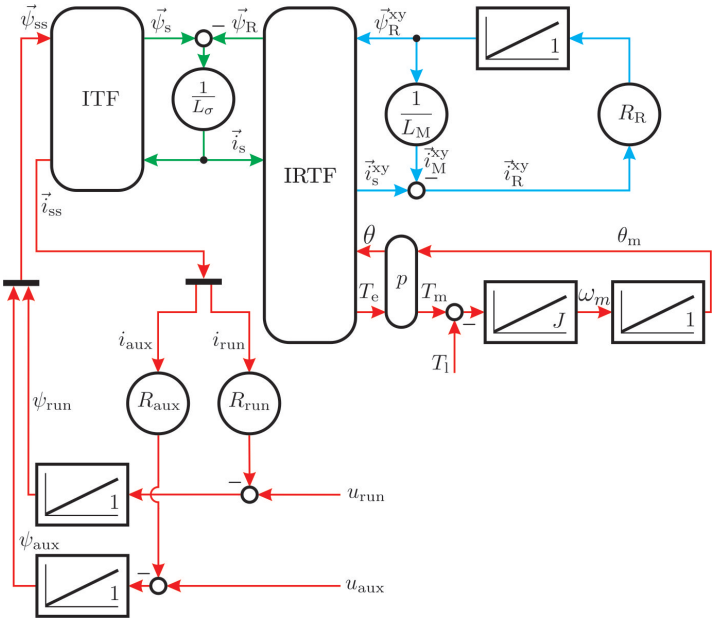


Fig. 2.28 Generic IRTF Induction machine model: single phase

distribution in the windings relative to the rotor flux, which implies the need for current control, as will be discussed shortly.

2.1.4 Current control for electrical drives

In this subsection we will briefly outline the basic principles of so called 'model based' current control [13] as applied in this GUIDE. An inductance/resistor type load is used in order to show how the controller parameters need to be chosen. Any machine can, in fact, be represented by such a network, as was discussed above (see figure 2.8) hence its use here. The use of 'model based' control in the so called synchronous reference frame is then discussed.

The current controller given in figure 2.29 shows a discrete PI control module, which has as inputs, the reference and measured currents from a load represented by an inductance and resistance. The converter is controlled (by the PI output) voltage source.

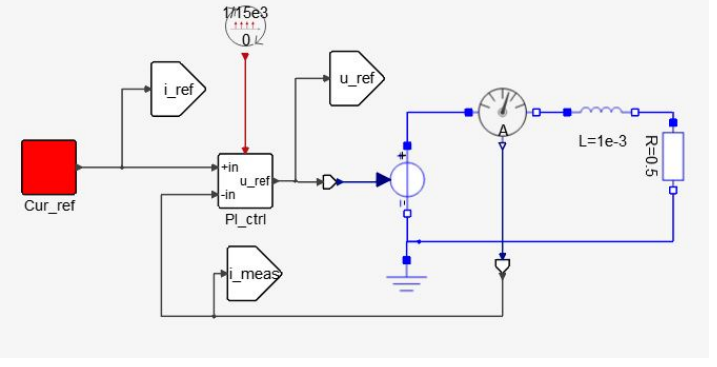


Fig. 2.29 Model based current control using an $R-L$ network to represent the motor

For regularly sampled control systems as considered here we are able to derive a simple controller structure which can be used to control the current in the $R-L$ circuit. Central to our proposed controller is the use of a modulator/converter combination, represented by the voltage source, which is designed to deliver in (a given sampling interval T_s) an average voltage per sample quantity $U(t_k)$, which corresponds to the set-point (reference) $U^*(t_k)$ value provided by the current control module. The task of the current control module is thus reduced to determining this set-point (reference) average voltage per sample value on the basis of the measured discretized load current $i(t_k)$ and the set-point current value i^*. The nature of this control philosophy is shown with the aid of figure 2.30. Shown in this figure are the non-sampled reference current $i^*(t)$ and typical (for PWM based control) converter current $i(t)$. These currents are sampled by the controller at time marks 0, t_1, t_2,.. etc.. At time $t = t_1$ a current error (between the sampled reference and sampled converter current) $i^*(t_1) - i(t_1)$ is present and the objective of the control approach is to determine the required average voltage reference value needed to zero said error. This leads to the condition $i^*(t_1) = i(t_2)$. The

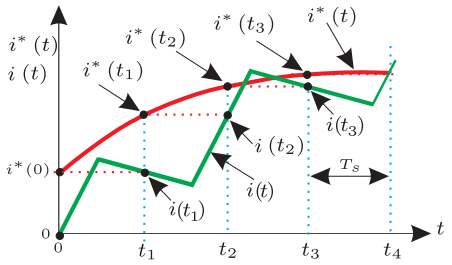


Fig. 2.30 Model based current control [14],[13]

control objective aimed at driving the current error to zero during each sample interval may therefore be written as

$$i\left(t_{k+1}\right) = i^*\left(t_k\right) \tag{2.25}$$

for a regularly sampled system with sampling time T_s.
The modulator/converter module will insure that the following condition is satisfied

$$U^*\left(t_k\right) = \frac{1}{\left(t_{k+1} - t_k\right)} \int_{t_k}^{t_{k+1}} u\left(\tau\right) d\tau \tag{2.26}$$

where u represents the voltage across the load (see figure 2.29). An observation of figure 2.29 shows that the load voltage may be expressed as

$$u = Ri + L\frac{di}{dt} \tag{2.27}$$

Use of equation (2.27) with equation (2.26) allows the latter to be written as

$$U^*\left(t_k\right) = \frac{R}{T_s} \int_{t_k}^{t_k+T_s} i\left(\tau\right) d\tau + \frac{L}{T_s} \int_{i(t_k)}^{i(t_k+T_s)} di \tag{2.28}$$

Discretization of equation (2.28), as discussed in subsection 1.4.3 allows the latter to be written as:

$$U^*\left(t_k\right) \cong R\, i^*\left(t_k\right) + \frac{L}{T_s} \Delta\, i\left(t_k\right) \tag{2.29}$$

with $\Delta\, i\left(t_k\right) = i^*\left(t_k\right) - i\left(t_k\right)$, which allows this expression to be written as

$$U^*\left(t_k\right) = R\, \Delta\, i\left(t_k\right) + \frac{L}{T_s} \Delta\, i\left(t_k\right) + R\, i\left(t_k\right) \tag{2.30}$$

which can also be written as

$$U^*(t_k) = K_p \, \Delta i(t_k) \, (1 + \omega_i \, T_s) + R \, i(t_k) \tag{2.31}$$

where K_p and ω_i as defined by equation (2.32) are known as the proportional gain and bandwidth of the current controller respectively [13].

$$K_{\mathrm{p}} = {}^{L}/_{T_s} \tag{2.32a}$$
$$\omega_i = {}^{R}/_{L} \tag{2.32b}$$

In practical applications the current control gain is typically set to $\frac{1}{4}$ of the value defined by equation (2.32a). Furthermore, the current controller bandwidth is constrained to $\omega_i^{\max} = 2000\, rad/s$ also for practical reasons.

In a drive, the modulator input is the modulation index defined as

$$m^*(t_k) = \frac{U^*(t_k)}{(U_{\mathrm{DC}}/2)} \tag{2.33}$$

where U_{DC} is the DC bus voltage. Use of this expression with equation (2.31) leads to

$$m^*(t_k) = K_p \, \frac{\Delta i(t_k)}{(U_{\mathrm{DC}}/2)} \, (1 + \omega_i \, T_s) + \underbrace{\frac{R \, i(t_k)}{(U_{\mathrm{DC}}/2)}}_{m_i(t_k)} \tag{2.34}$$

where $m_i(t_k)$ is the modulation index value generated by the integrator alone, sampled prior to the current value. The analysis carried out to this point was done with a floating point based controller. In practice a fixed point controller is ofter used hence scaling is needed , as discussed for the speed controller. This implies that scaling, as discussed in subsection 1.4.3, needs to be undertaken for equation (2.34) which leads to

$$m^*(t_k) = \underbrace{\left(\frac{i_{fs} \, K_p}{u_{fs}}\right)}_{\text{Gain } \mathrm{K_p}} \frac{2\Delta i_n(t_k)}{U_{\mathrm{DC}}} \left(1 + \underbrace{\omega_i \, T_s}_{\text{Gain } \mathrm{w_i}}\right) + m_i(t_k) \tag{2.35}$$

The Embed current controller example given in figure 2.31 shows the PI control based on equation (2.35) and gain settings defined by equation (2.32). The `udely` module is used to generate the $m_i(t_k)$ term. A series network consisting of an inductance/resistance network acts as a load, which is connected to a 'converter' module which generates a voltage output defined as $U^{\text{ref}} = m^{\text{ref}}\, U_{\text{DC}}/2$. Said module also includes a limiter which constrains the output to limits $\pm U_{\text{DC}}/2$.

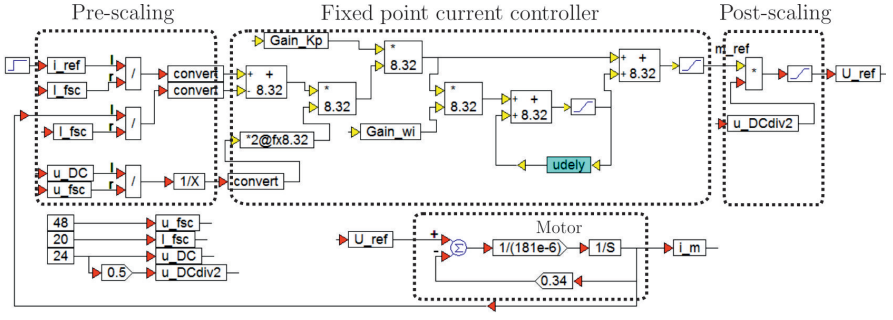


Fig. 2.31 Embed, fixed point Model based current control example [14]

In practical implementations, the PI controller in its current form is prone to 'windup', which occurs when the limits of a linear system are reached. Windup can occur in this case when the reference average voltage value generated by the PI controller exceeds the maximum value which can be delivered by the converter. Under such circumstances a current error occurs at the input of the controller which will cause the integrator output to ramp up or down. Practical controllers have an 'anti-windup' feature, as shown in figure 2.31 by way of the `lb`,`ub` limit modules, which constrains integrator action when the user defined limits are reached. The L, R parameters used in figure 2.31 are set to $L = 0.181$ mH and $R = 0.34\,\Omega$. A typical controller step response under locked rotor conditions with a DC bus voltage of 24 V and 1 kHz sample frequency $\left(\frac{1}{T_s}\right)$ is given in figure 2.32.

In three-phase drives, both magnitude and orientation of the current vector $\vec{i}$ relative to the 'd,q' reference frame must be controlled, which requires the use of the conversion module, as shown in figure 2.33. This module initially transforms the measured currents into a vector with components i_α, i_β , using a 'Forward Clarke' transformation. This vector is however, referenced to a stationary reference frame, i.e tied to the stator of the motor. Hence an additional transformation, using a 'Forward Park' is needed to transform

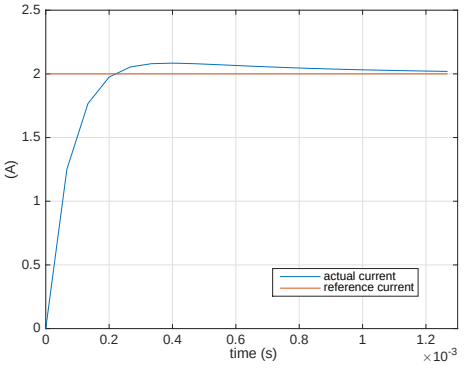


Fig. 2.32 Step response with Embed, fixed point model based controller and L, R load

the vector $\vec{i}$ to the synchronous 'd,q' reference frame, as required for torque control.

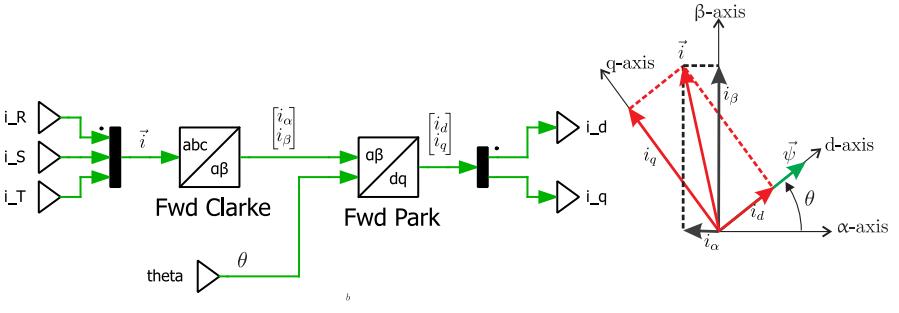


Fig. 2.33 Current conversion to 'd,q' coordinates

Input to this module are the three (in most cases, but two is also possible) measured currents and the angle θ, which defines the orientation of the 'd-axis'. When current control is used for field-oriented control (FOC) said axis must be aligned with the magnetic flux vector $\vec{\psi}$. In a sensored drive it is provided via an encoder. In a sensorless drive, this angle is calculated, using either an InstaSPIN or AFT (Altair Flux Tracking) DLL based algorithm.

Current control, in this so called synchronous reference frame, requires two model based current controllers, as shown in figure 2.34.
Note that in practical examples the output of a synchronous controller is defined by a user specified maximum voltage $u_{\max}$ (or maximum modulation

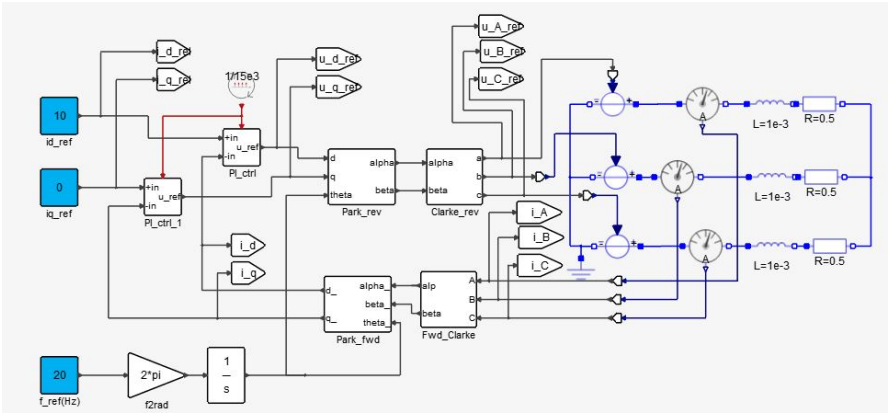


Fig. 2.34 Synchronous current control of a machine represented by an 'R-L' network

index), hence a circle. The direct axis output u_d^{ref} is used to constrain the quadrature controller output using:

$$u_q^{\text{ref}} = \sqrt{u_{\max}^2 - \left(u_d^{\text{ref}}\right)^2} \tag{2.36}$$

The output of these two controllers are the u_d^{ref}, u_q^{ref} values, which must be transformed to the stationary (α, β) motor reference frame using a 'Reverse Park' module, as shown in figure 2.34. Finally a 'Reverse Clarke' transformation is used to generate the three reference phase values for the voltage sources `u_A`, `u_B` and `u_C`. For electrical machines the 'L-R' load (shown as `L`, `R` in figure (2.34) is predominantly inductive, i.e. $^{L}/_{T_s} > R$ in which case the voltage vector can be represented in the synchronous reference frame as

$$\vec{U}^* \cong \frac{L}{T_s}\left(\vec{i}^* - \vec{i}\right) \tag{2.37}$$

where $\vec{i}^* = i_d^* + j i_q^*$ is the reference current vector, set by the user, while $\vec{i}$ represents the measured current.
The task of the current controller is therefore, to generate the voltage vector needed to reduce the error between these currents to zero during each sampling interval T_s. An example which shows the relationship between current error and voltage vector is given in figure 2.35.

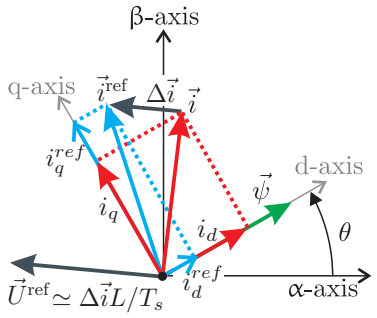


Fig. 2.35 Required voltage vector needed to minimize a current error $\Delta\vec{i}$

The task of the modulator/converter is to generate a voltage vector which matches the reference vector, as will be discussed in the following subsection.

2.1.5 Space Vector modulation and Timing Issues

In the example shown in figure 2.34 the reference voltage vector was converted to three phase voltages `u_A_ref`, `u_B_ref` and `u_C_ref` which were then used as inputs to a converter represented by linear amplifiers. In reality, use is made of a converter, as shown in figure 2.36, which consists of a PWM unit that controls the switching converter. This converter `Sw_Conv` is conveniently represented by six switches that are controlled by logic signals a, b, c as shown in figure 2.37. The converter is connected to a DC supply u_{DC}, whilst the outputs A, B, C of said module are connected to a motor, which has been conveniently represented by three resistors-inductances. The voltage across the motor phases u_A, u_B, u_C will depend on the status of the switches and can be either $\pm^2 u_{\mathrm{DC}}/3$, $\pm u_{\mathrm{DC}}/3$ or zero. In the latter case the outputs of the converter are simultaneously connected either the DC bus supply u_{DC} or the $0\,V$ terminal. The modulator called `PWM_unit`, also shown in figure 2.36, generates the desired logic signals that control the converter switches. The chosen logic sequence of the signals a, b, c (see figure 2.37) and the duration for which these should be active is determined by the three phase reference signals `u_A_ref`,`u_B_ref` and `u_C_ref` . These input signals are scaled by half the DC bus voltage $u_{\mathrm{DC}}/2$, in which case they are referred to as modulation indices : $m_{\mathrm{A}}^{\mathrm{ref}}$, $m_{\mathrm{B}}^{\mathrm{ref}}$, $m_{\mathrm{C}}^{\mathrm{ref}}$, which are connected to the pulse width modulation unit that generates the three logic signals a, b, c for the switches. When a gate signal is '1' the corresponding motor terminal is connected to the DC supply source. Vice-versa a logic '0' implies that the motor terminal output is connected to the $0\,V$ of the DC supply.

Only a limited number of logic signal a, b, c combinations can be generated by the converter and these can be represented by six active and two

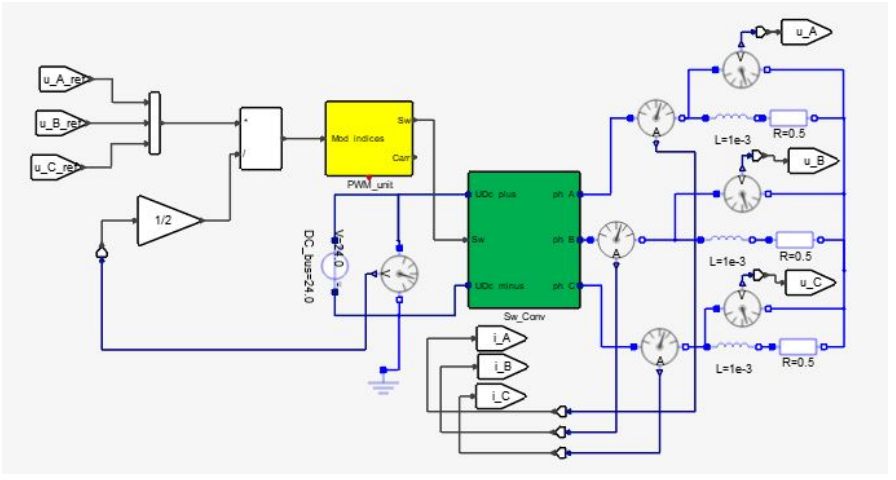


Fig. 2.36 Converter/modulator

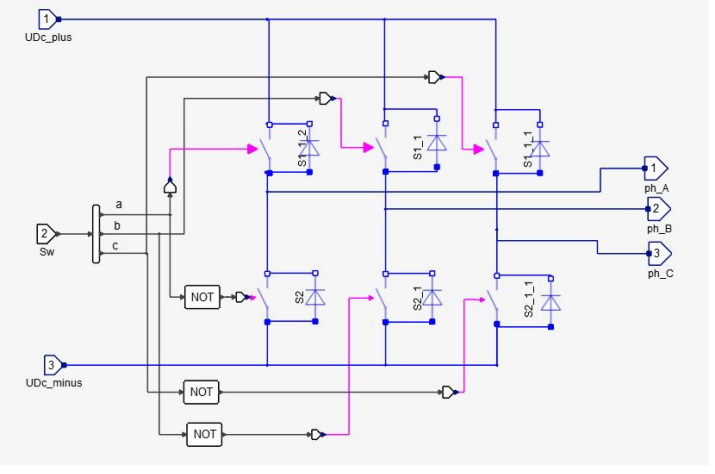


Fig. 2.37 Ideal switching Converter

'zero' output voltage vectors as shown in figure 2.38. These output vectors are generated by considering the voltages across the three motor phases for a given logic sequence $a,\ b,\ c$, and then applying a forward Clark transformation. Each vector, with amplitude $2u_{DC}/3$ [13], is shown in terms of the logic signal combination $a,\ b,\ c$, hence $\vec{u}_{120}(010)$, implies $a = 0$, $b = 1$ and $c = 0$. A 'zero' vector denotes the conditions where the output voltage will be zero, i.e. all switches connected to the positive or zero bus voltage terminal.

The task of the modulator is to identify the two adjacent vectors (as shown in figure 2.38) which must be used to achieve a given average voltage per

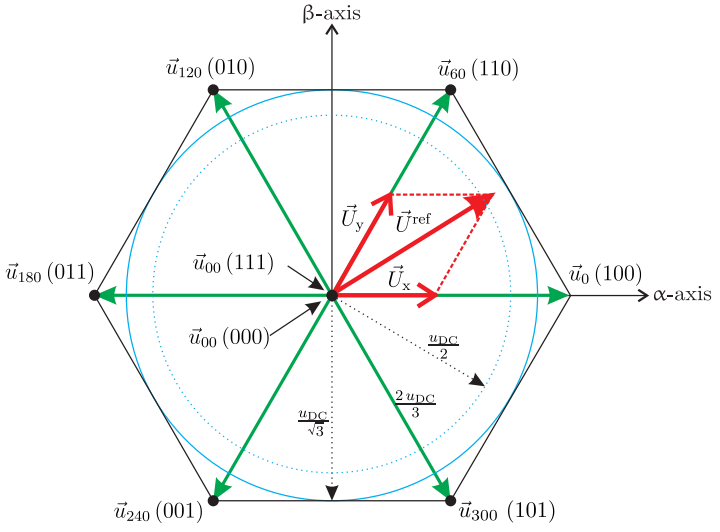


Fig. 2.38 Available converter output vectors [13]

sample $\vec{U}^{\text{ref}}(t_k)$ and the PWM time interval T_{PWM} in which case either one or two active vectors must be selected. In the example shown the motor output voltage vector will be of the form;

$$\vec{U}^{\text{ref}}(t_k) = \underbrace{\frac{\Delta T_x}{T_{\text{PWM}}}\vec{u}_0}_{\vec{U}_{\text{x}}} + \underbrace{\frac{\Delta T_y}{T_{\text{PWM}}}\vec{u}_{60}}_{\vec{U}_{\text{y}}} \tag{2.38}$$

where the time interval variables ΔT_x, ΔT_y are found by comparing the duty cycle values defined as:

$$d_i = {}^{(m_i^*+1)}/_2 \tag{2.39}$$

for $i = 1, 2, 3$ with a triangular waveform (amplitude '1') as shown in figure 2.39.

Observation of figure 2.39 shows that the required time variables are evenly distributed in each **half** of the PWM sample interval T_{PWM}. The variables PWM_1, PWM_2, PWM_3 are logic signals, which are equal to the signals 'a','b','c' that control the converter switches (see figure 2.37). Note that the sum of ${}^{(\Delta T_x+\Delta T_y)}/_2$ must be less or equal to ${}^{T_{\text{PWM}}}/_2$, hence if it is

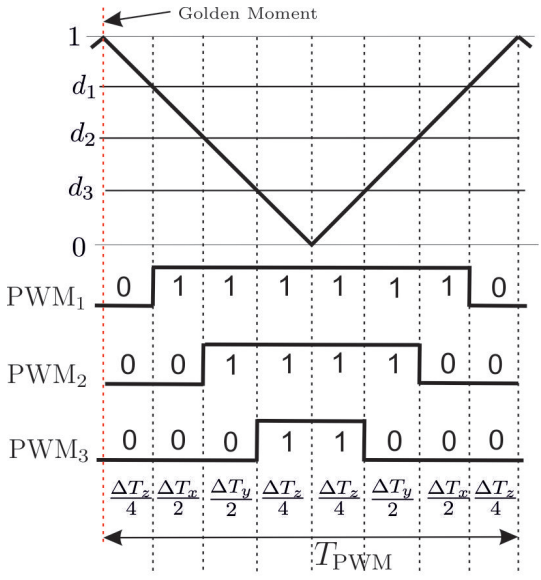


Fig. 2.39 PWM generation

less, the time interval $\Delta T_z/2 = T_{\mathrm{PWM}}/2 - (\Delta T_x + \Delta T_y)/2$ must be complemented by a convenient choice of zero vectors . So called 'third harmonic' injection or 'pulse centering' [13], distributes the available time $\Delta T_z/2$ to either side of the interval $T_{\mathrm{PWM}}/2$, as shown in figure 2.39. The use of pulse centering increases the available linear operating range to a circle with radius $u_{\mathrm{DC}}/\sqrt{3}$. Without pulse-centering the operating range is limited to $u_{\mathrm{DC}}/2$, hence a factor $2/\sqrt{3} \approx 1.15$ increase is achieved by using this technique. Note that the start of the pulse-width modulation period T_{PWM} is referred to as the 'GOLDEN MOMENT 'as it represents a point in time where all three lowest switches of a converter (see figure (2.2) are 'on', which is important given that the shunt resistors shown in said figure can then measure the phase currents.

In the sequel to this section we will discuss the timing aspects, which basically describes at which time interval various activities such as data acquisition, defining the active set of converter vectors and execution of various control algorithms take place. This discussion is carried out with the aid of figure 2.40, which (among others) shows the CPU clock ($\mathrm{CPU}_{\mathrm{clk}}$), in this case set to 90 MHz, that is connected to an up/down counter (TBCTR), that replaces the triangular waveform shown earlier in figure 2.39. The start of each PWM period time, known as the 'GOLDEN MOMENT' coincides with the highest counter value TBPRD , and its value is therefore determined by

$$\text{TBPRD} = \frac{T_{\text{PWM}}}{2}\text{CPU}_{\text{clk}} \tag{2.40}$$

hence a PWM frequency of 10 kHz, which corresponds to T_{PWM} =0.1 ms, requires a maximum counter value of TBPRD=45000. The required ON/OFF times of the power switches for a given converter leg is found by comparing the up/down counter value with the discrete duty cycle value d_c which is now defined as

$$d_c = \text{TBPRD}\frac{(m_i^*(t_k)+1)}{2} \tag{2.41}$$

where m_i^* is the reference (command value) modulation index for each phase $i = 1, 2, 3$.

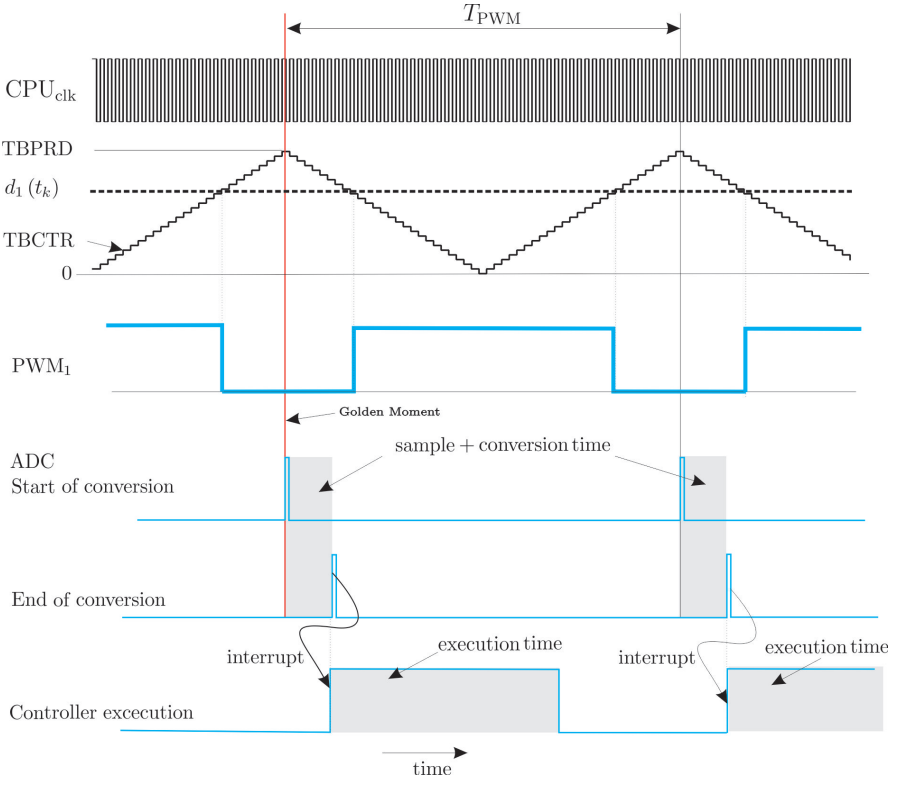


Fig. 2.40 Timing issues[23]

At the beginning of each PWM interval (the GOLDEN MOMENT) the ADC start of conversion pulse, activates the ADC (Analog Digital Conversion) data acquisition phase, where all the currents and voltages are sampled. If using current shunts, measurement at the GOLDEN MOMENT must be done. However, when deploying in-line phase measurements units like LEM's, this is less critical and also gives more flexibility for operation at high modulation indices (approaching so called 'six-stepping').

The readers attention is drawn to the fact that a potential timing problem can occur when using shunts (connected to the lower converter switches) , given that the ADC sampling process takes a finite amount of time, defined by the difference between the start of ADC converter (beginning of T_{s}) and the ADC 'End of Conversion' signal. During this time interval the bottom switches must be ON, which ensures that the shunts measure the phase currents. If however the modulation index d_1, shown in figure 2.39 increases, then the time interval ΔT_z and (in this example) the PWM_1 signal will exhibit a smaller time interval where a bottom switch is ON. Consequently, a situation can occur where ΔT_z is such that the bottom switch may not be ON whilst ADC sampling occurs. In that case the ADC result will be erroneous. Under these conditions we can replace the erroneous output, by realizing that the sum of the three phase currents must be zero (if the motor star point is not connected, as is usually the case). Hence we can replace the faulty signal by using the two other ADC current signals . Hence it is preferred to use three current shunt sensors (if shunts are chosen for low cost applications), if operation with large modulation index values is required (as is nearly always the case). However, the use of in-line phase measurement (with three sensors) is always recommended, when compared to using shunts located underneath the lower converter switches, given that no ADC sampling issues for large modulation values will be encountered. Furthermore, the use of in-line phase measurement gives the option of using 'oversampling', which implies that multiple current and voltage samples at a rate typically 4 to 8 times the controller sampling frequency can be obtained and then averaged prior to use by the control algorithm.

The 'End of Conversion' signal , triggers an 'EOC' interrupt which signifies that data is ready for processing by a control algorithm. The control algorithm will calculate the reference modulation indices m_i^*, for $i = 1, 2, 3$ for the next time interval. Note that the EOC interrupt can be used to trigger the execution of a control algorithm during each cycle, or a number of cycles. For example, the following timing options can be used:

- each EOC pulse: Trigger the main controller algorithm which gives a sampling time $T_s = T_{\mathrm{PWM}}$. This will work provide the control execution time is LESS then T_{PWM} and is in the approach illustrated in figure 2.40.

- every second EOC pulse: Trigger the main controller algorithm which gives a sampling time $T_s = 2\,, T_{\mathrm{PWM}}$. Therefore doubling the time control execution time and implies that the controller sampling frequency is half of the PWM frequency. T_{PWM}.
- every third EOC pulse: Trigger the main controller algorithm, which gives a sampling time $T_s = 3\,T_{\mathrm{PWM}}$. This implies that the sampling frequency is one third of the PWM frequency.

This so called segmenting of various control algorithm's is needed to ensure that execution times stay within the time available. This issue becomes particularly important when very high PWM frequencies are used because the motor time constant $\frac{L_s}{R_s}$ is low (as is the case for drone applications), which leads to high current ripple and ADC measurement problems (in particular when using shunts).

2.1.6 eDRIVES Example structure

In the ensuing chapters a detailed overview is given of the eDRIVE examples with the primary purpose of reducing the development time needed to arrive at a drive that needs to be implemented by a user. The basic example structure, as given in figure 2.41, is initially based on the motor configuration that the user needs to deploy. In addition, this primary category has a special 'dual drive' category, specifically designed for dual machine applications.

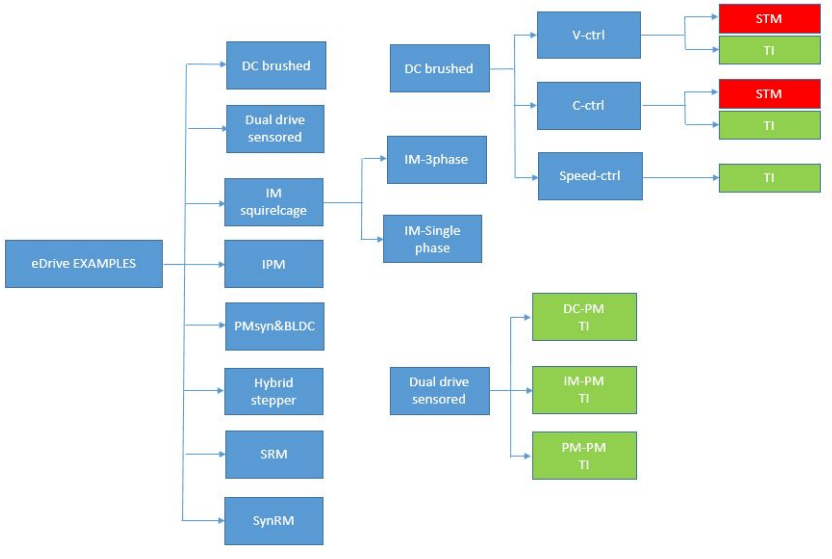


Fig. 2.41 Primary eDRIVE example structure

After selection of a given category, e.g 'DC brushed', a set of subcategories is available which is based on the nature of the control algorithm. For 'brushed DC' the options are:

- V-ctrl:armature voltage control
- C-ctrl:armature current control
- Speed-ctrl:armature speed control, where use is made of an encoder signal from an attached PM machine

The category induction machines (IM squirrel cage) refers to machine with a squirel cage rotor and the ensuing subdivision is 'three-phase' and 'single-phase'. The latter is in fact a two -phase machine and control is similar to that of a three-phase machine. A three-phase inverter is used to achieve this. The subcategories for IM, as shown in figure 2.42 are again focused on the control algorithms that can be implemented. For the three-phase IM the following

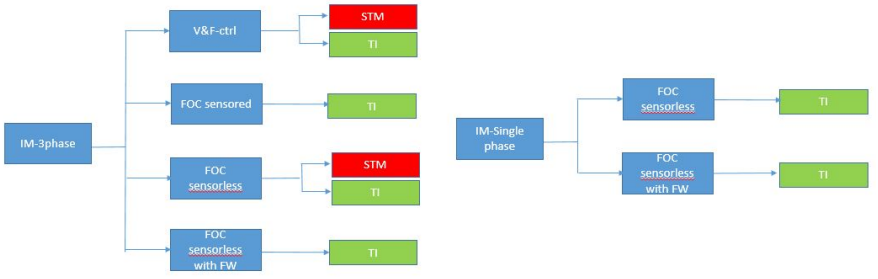


Fig. 2.42 IM eDRIVE example stucture

control options are available:

- V&F-ctrl: voltage and frequency control, which means generating a rotating voltage vector with a rotational speed set by the frequency.
- C& F-ctrl: current and frequency control, which means generating a rotating current vector with a rotational speed set by the frequency.
- FOC-sensored: Field-oriented control where the current vector is oriented relative to the rotor flux vector and this requires access to a shaft speed encoder. In this case provided by the encoder of a PM machine, attached to the IM.
- FOC-sensorless: Field-oriented control where the current vector is oriented relative to the rotor flux vector and here use is made of an encoderless algorithm such as 'Altair Flux Tracker' (AFT) or Texas Instruments FAST. This approach requires access to the measured converter voltages and currents. In addition, parameter knowledge is required.

- FOC-sensorless with FW: Use of a sensorless control algorithm with a added component called 'field-weakening' (FW) designed to reduce the rotor flux at higher speeds.

For single phase IM, the following options are available:

- FOC-sensorless: Field-oriented control where the current vector is oriented relative to the rotor flux vector and use is made of an encoderless algorithm's such as 'Altair Flux Tracker' (AFT) or Texas Instruments FAST. This approach requires access to the measured converter voltages and currents. In addition, parameter knowledge is required.
- FOC-sensorless with FW: Use of a sensorless control algorithm with a added component called 'field-weakening' (FW) designed to reduce the rotor flux at higher speeds.

Note that the option 'FOC' is usually combined with 'current & frequency' control, so that the user can check operation of the machine using said approach first, before moving on to FOC.

The category option 'PMsyn&BLDC', shown in figure 2.43 has the following control categories:

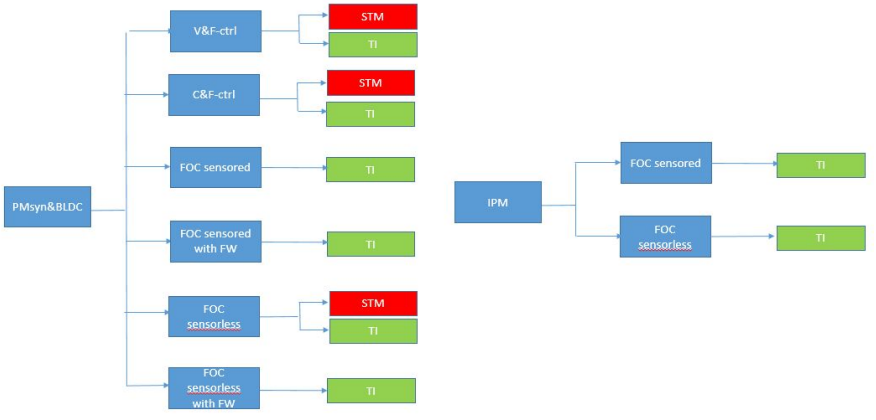


Fig. 2.43 PMsym&BLDC eDRIVE example stucture

- V&F-ctrl: voltage and frequency control, which means generating a rotating voltage vector with a rotational speed, set by the frequency. This subcategory is often the starting point for drive development as it is used to test the hardware and sensor data.
- C& F-ctrl: current and frequency control, which means generating a rotating current vector with a rotational speed set by the frequency.
- FOC-sensored: Field-oriented control where the current vector is oriented relative to the rotor flux vector and this requires access to a shaft speed

encoder or Hall transducers. Often, these examples also include 'current and frequency control', to simplify the transition to FOC.
- FOC-sensored with FW: Field-oriented control where the current vector is oriented relative to the rotor flux vector and this requires access to a shaft speed encoder. Added to these examples is 'field-weakening', which implies manipulation of the direct axis current component to reduces the effective terminal voltage amplitude. This is subject to the short-circuit current amplitude, relative to the maximum motor current as will be discussed in the relevant examples.
- FOC-sensorless: Field-oriented control where the current vector is oriented relative to the rotor flux vector and here use is made of an encoderless algorithm such as 'Altair Flux Tracker' (AFT) or Texas Instruments FAST. This approach requires access to the measured converter voltages and currents. In addition, motor parameter knowledge is required.
- FOC-sensorless with FW: Use of a sensorless control algorithm with a added component called 'field-weakening' (FW) designed to reduce the effective terminal voltage at higher speeds.

The three categories shown in figure 2.44 are concerned with the following motor topologies:

- Hybrid stepper: which can be viewed as a two-phase multipole BLDC, which is connected to a three-phase converter. These applications use sensorless FOC and is usually combined with 'current -frequency control'.
- Switched Reluctance (SRM): these examples are based on the use of a four phase SRM that is connected to a 3-phase converter. The control algorithms include 'current-frequency control and FOC, both sensored and sensorless.
- Synchronous Reluctance (SynRM): FOC sensorless control, together with 'current&frequency' control, are examples that will be discussed.

The examples shown in the 'green'and 'red' boxes (see figures 2.41,2.42,2.43 and 2.44) referred to as 'TI'and 'STM' respectively are categorized in terms of the hardware implementations as indicated below (see figure 2.45). Each hardware category is linked to a 'target', which represents the drive tool chain typically used for drive development.

- Simulation (SIM): Drive simulation with a discrete, fixed point representation of the control module designed for use with an MCU. In this phase the converter switching process is not modelled. Instead, the average voltage per sample is generated using the product of modulation indices (calculated by the controller) and half the DC bus voltage. Its primary purpose is to verify fixed point controller operation of the drive.

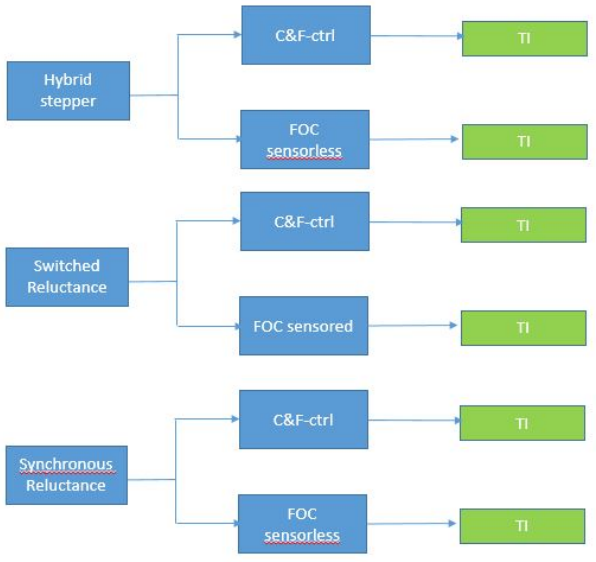


Fig. 2.44 Hybrid stepper, SynRM and SRM eDRIVE example stucture

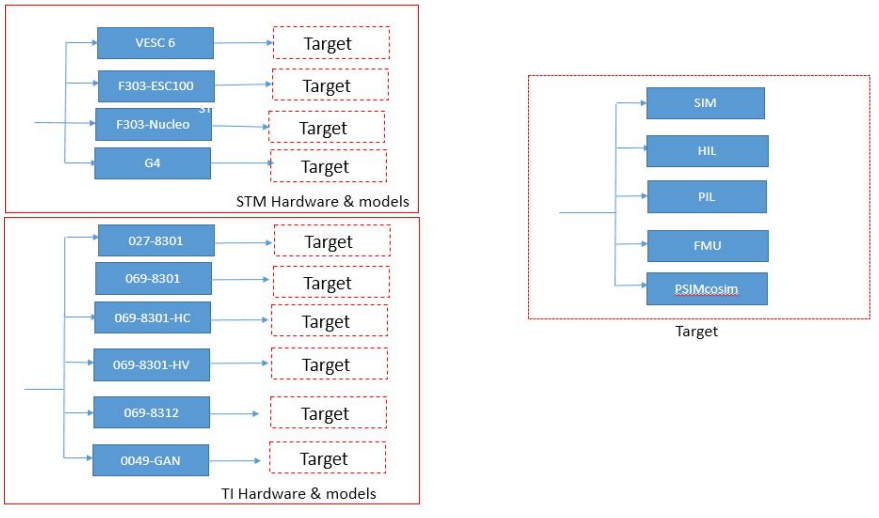


Fig. 2.45 Hardware and target categories, eDRIVE example structure

- Processor in the loop (PIL): This step is needed for sensorless examples, as use is made of an algorithm like 'AFT' or 'FAST', which cannot be evaluated as a 'simulation'. However, the PIL approach is also instructive for other examples, where the user can verify that the control functions as expected with the MCU operating on the target board. The motor and controller used here, are as indicated in the previous item.
- PSIMcosim: In examples where it is important to examine the impact of using a switching converter, this option is required. This implies the use of a controller as present in the SIM or PIL stage and combining this with 'co-simulation' in PSIM. In PSIM a switching converter is used together with a machine model as deployed in the previous two items.
- Hardware in the loop (HIL): This phase of development uses a discrete, fixed point representation of the control module as discussed above and uses the hardware of both converter and machine. This stage of development has two components namely:
  - HIL-compile: MCU version of the controller, using Embed software. In this phase, the additional building blocks such as ADC, PWM and diagnostic buffers (used to examine data in the next phase) are added to the module, which is then compiled and linked with the Texas Instruments and Embed libraries, which leads to an 'out-file' that must be down-loaded to the MCU via the JTAG connection. The file can reside in either MCU RAM, or FLASH memory.
  - HIL-debug: Run version of HIL, using Embed software. In this phase the user can execute the 'out-file' file created in the 'compile' phase and view outputs whilst controlling the drive. This out-file is downloaded to the MCU prior to running the example. If using MCU RAM this download is carried out via Embed, otherwise (when using MCU Flash memory) Texas Instruments Uniflash can, for example, be used.

In the series of examples which will be considered in the ensuing chapters, not all development stages of the tool-chain will necessarily be invoked. In some cases a detailed inverter switching analysis, may, not be required, which means that the COSIM stage is then omitted. During the HIL debugging stage it is often necessary to cycle between components compile → debug → compile where internal variables are visualized on a diagnostic scope.

For examples which require the use of a single machine, the non-essential machine is mechanically removed. However, when considering induction machine or brushed DC operation a PM machine is mechanically connected in order to use the encoder that is fitted to the machine. A 24V/4A DC power supply unit is used for most examples. However, there are examples which operate at higher bus voltages. A DC current probe and oscilloscope both of which are recommended to experimentally verify results.

In terms of software implementation, a laptop or PC is required, which contains the following software packages:

- EMBED: used to do real time control and simulation using fixed point controllers . The eDRIVES software package is required to run the examples discussed in the ensuing chapters.
- PSIM:used for the co-simulation component in the toolchain mentioned above.
- Texas Instruments: UniFlash in the event that files need to be transferred to the RAM of the MCU (applies to TI boards).
- ACTIVATE: used above to show basic drive principles.

Chapter 3
Brushed DC drives

This chapter is concerned with the control of brushed DC drives and as such the developmental tool-chain shown in figure 2.45 is followed. This implies that in most cases the toolchain: 'Simulation' (SIM), 'Processor in the Loop'(PIL),'CO-simulation' (COSIM) and 'Hardware in the Loop'(HIL) is discussed for Texas Instruments (TI) and ST-Micro (STM) based hardware examples. The drive control options for brushed DC motors are: 'Armature Voltage Control','Armature current Control' and 'Speed control'. In addition, multiple HIL examples are introduced with different hardware, to reduce drive development time.

3.1 Armature Voltage Control

In figure 2.29 a single resistive inductive load is shown which is characteristic for a brushed DC machine (see figure 2.10). In the figure, a voltage source is introduced, which in practical applications must be replaced by a 'switching converter', as shown in figure 3.1. This type of converter (commonly referred to as a 'H' bridge) consists of two half-bridges with outputs `ph_A`, `ph_B`, which are connected to the armature of a brushed DC machine. The voltage across the armature, u_A is therefore defined by the two converter output voltages, i.e the difference between these two, as is apparent from figure 3.2. Readily apparent in figure 3.2 is the 'H-bridge' `Sw_Conv`, which is connected to a module `PWM unit` that controls the switching process using pulse-width modulation (PWM) as introduced in section 2.1.5 for three-phase machines. The purpose of the ACTIVATE model (see figure 3.2) is to show converter switching and this makes use of a simplified electrical load representing a locked rotor brushed DC machine.

For brushed DC machines, considered here, the modulation approach is similar to that shown for three-phase machines and in this case the reference

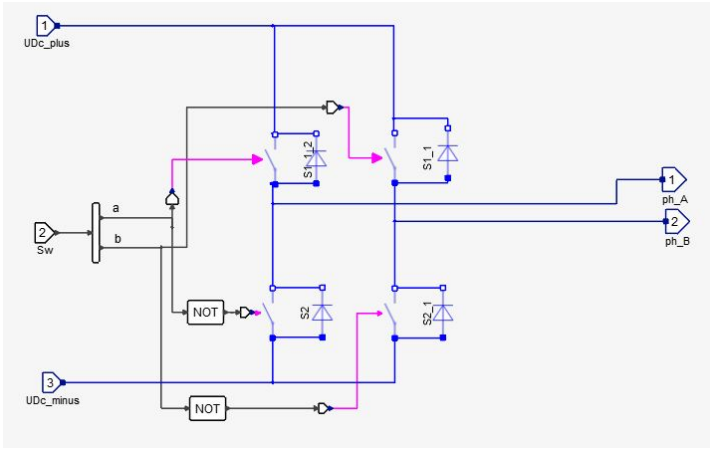


Fig. 3.1 'H-bridge' converter

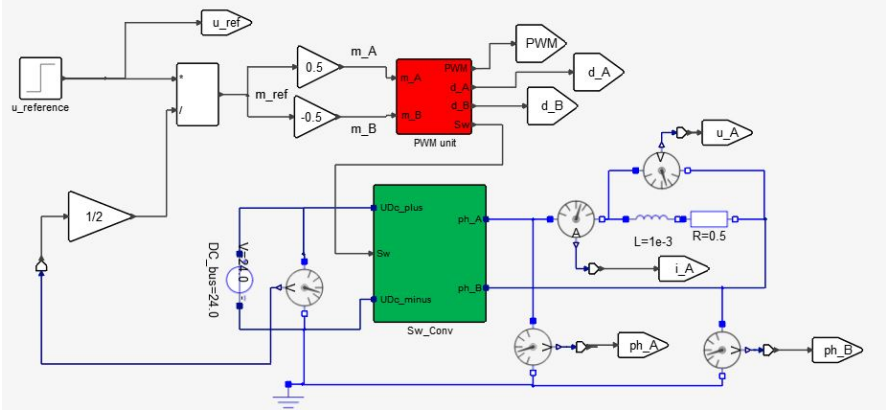


Fig. 3.2 Voltage control using PWM for a Brushed DC machine

voltage `u_ref` is converted to the modulation index using ${}^{2\,u_{ref}}/_{u_{DC}}$, where u_{DC} represents the DC bus voltage of the converter. For single phase voltage control, as required for brushed DC drives, the reference modulation index defines the indices m_A, m_B using $m_A = {}^{m_{ref}}/_2$ and $m_B = -{}^{m_{ref}}/_2$ respectively. These modulation indices are in turn used to define the duty cycle values `d_A`, `d_B` used by the `PWM unit` (see equation 2.39).

Results obtained using the ACTIVATE®model (figure 3.2) serves to demonstrate how armature voltage is implemented.

The PWM switching process, as given by figure 3.3 shows the PWM signal, together with the two converter duty cycles. For this example, the reference

voltage is step function that is initially zero, and at 0.5 ms switches to 10 V. Also shown in this diagram are the outputs of the two half bridges `ph_A`, `ph_B` and the voltage across the armature `u_A`. A further illustration of voltage

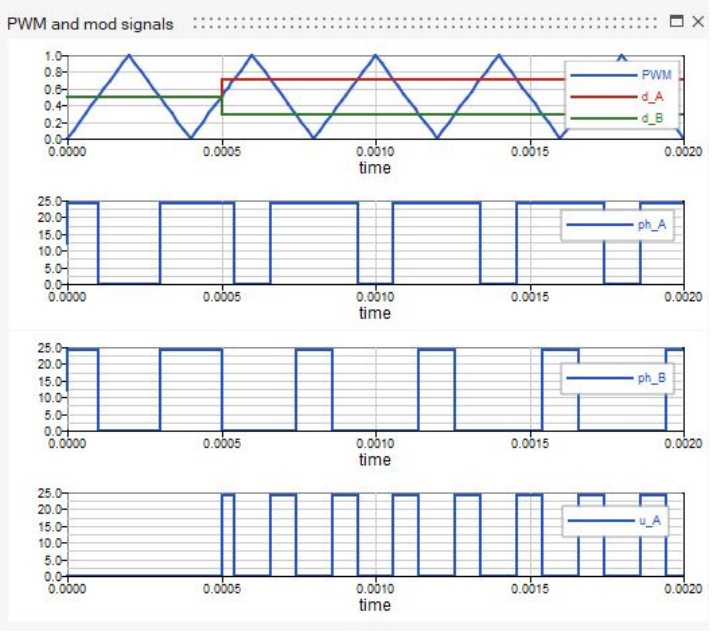


Fig. 3.3 PWM switching process and converter/armature voltage outputs

control is given in figure 3.4, which shows the reference voltage `u_ref`, the voltage across the armature due to the switching converter `u_A` and the armature current over a 2 ms simulation time. Note that this example assumes that the armature is mechanically locked, i.e the electrical circuit is simply the resistance and inductance, as mentioned earlier.

In the ensuing subsection a detailed set of examples is given which is based on the principle outlined above.

3.1.1 Armature Voltage Control-TI

Purpose of this section is to realize open loop armature voltage control of a brushed DC machine using Texas Instruments hardware. At the same time this type of example is used to make a functional check on the current/voltage measurements and overall functioning of the drive board and software. The

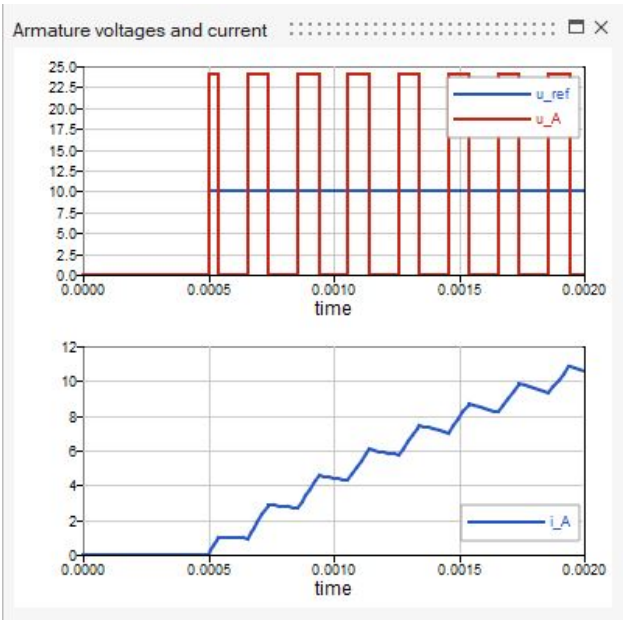


Fig. 3.4 Armature voltage (switching and reference) /current of a Brushed DC machine

tool chain as introduced in figure 2.45 is used as a template for the ensuing subsections.

3.1.1.1 Voltage control-F069-boostXL8301: SIM

Reference file: `BrushDC-VControl-F28069M-BoostXL-DRV8301-SIM.vsm`

This example considers the simulation of a brushed DC motor [25] connected to a Texas Instruments F069 Launchpad, with a three-phase boostXL8301 converter, as shown in figure 3.5. The aim is to examine operation of the drive using a fixed point controller, that will subsequently be used in the HIL implementation. The motor is connected to phases 'A' and 'B', hence 'C' is not used in the example.

Drilling into the 'voltage controller' reveals the controller structure shown in figure 3.6. Outputs of the controller are the two modulation indices that control the two converter half bridges in use. For all controllers considered, the aim is to generate the modulation indices that are required by the PWM units. Consequently, it is important to convert the reference voltage to the modulation index using expression 3.1.

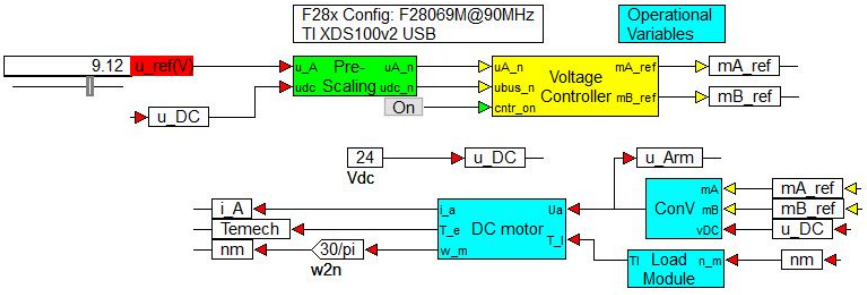


Fig. 3.5 Brushed DC drive with armature voltage control

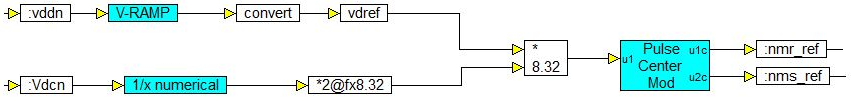


Fig. 3.6 Voltage controller structure

$$m^{\text{ref}} = \frac{2\,u^{\text{ref}}}{u_{\text{DC}}} \tag{3.1}$$

which requires access to the measured DC bus voltage u_{DC}, shown as `u_DC`. The 'Pulse Center Mod' unit splits the reference modulation index into components $\pm \frac{m^{\text{ref}}}{2}$ as required to modulate the two half-bridges used. The voltage supplied to the motor `U_Arm` is found by multiplying the modulation indices `mA_ref`, `mB_ref` by half the DC bus voltage as shown in figure 3.5 (using the `ConV` module) and taking the difference between the two half bridge voltages. These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables $u_a^{\text{ref}}(t_k)$, $u_b^{\text{ref}}(t_k)$.

The aim of the switching converter is to produce the required average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sections 2.1.5, 3.1). In this case the armature is directly connected between two converter phases and this implies that the maximum modulation index is equal to $m_{max} = 2$, which corresponds to an armature voltage of 24V. A quadratic load is introduced in the load module dialog box, which provides a value of 100 mNm at a shaft speed of 1000 rpm.

The results shown in figure 3.7, confirm the presence of armature volt-

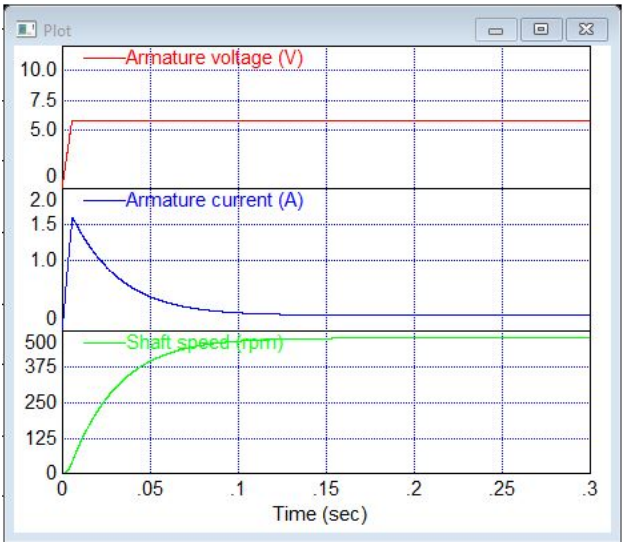


Fig. 3.7 Simulation (SIM) result, brushed DC operation, voltage control

age control. Note that at start up a ramp function is used, which determines the rate of reference variable change, as set in the 'operational Variables' dialogue box. The steady state shaft speed is governed by the armature voltage reference setting (set to 5.28 V in this example) and the applied load characteristic.

3.1.1.2 Voltage control-F069-boostXL8301: PIL

Reference files:

- `BrushDC-VControl-F28069M-BoostXL-DRV8301-PIL.vsm`
- `DC_Vctrl_PIL.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F28069. This implies that the controller module shown in the simulation (see figure 3.6) is now replaced with a file: `DC_Vctrl_PIL.out` as indicated in figure 3.8. The target interface model that executes the out-file, generates

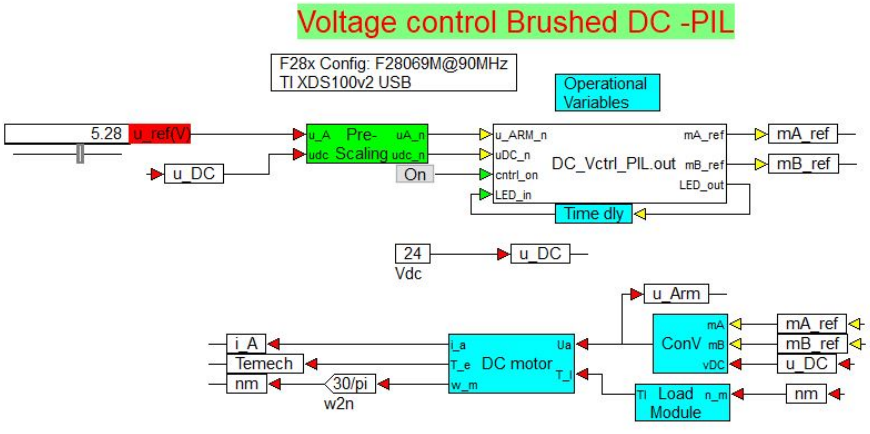


Fig. 3.8 PIL based drive, brushed DC operation, voltage control

the modulation indices required for the converter motor, and these components are not affected by the use of the PIL. Generation of the out file is based on the PIL module, see figure 3.9, which must be compiled using the command: `toolbox>codegen`. Moving one level deeper into the PIL module

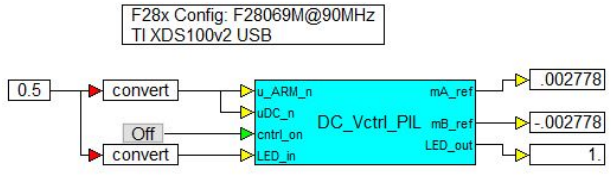


Fig. 3.9 PIL module, brushed DC operation, voltage control

`DC_Vctrl_PIL` shows the voltage controller (see figure 3.10) as introduced in section 3.1.1.1.

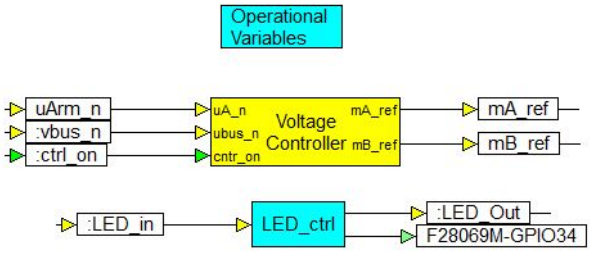


Fig. 3.10 Content of the PIL module, brushed DC operation, voltage control

Added to the PIL voltage controller is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that PIL control is operational.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational. Results produced by PIL operation are identical to the plot shown in figure 3.7 and are therefore not shown here.

3.1.1.3 Voltage control-F069-boostXL8301:COSIM

Reference files:

- `BrushDC-VControl-F28069M-BoostXL-DRV8301-COSIM.vsm`
- `BOOSTXL_DC_mot.psimsch`
- `DC motor model_v1.psimsch`

To evaluate operation of a controller with a switching converter, in this case the `boostXL8301`, use can be made of a `PSIMCoupler` module, as indicated in figure 3.11. This approach requires the use of PSIM together with Embed, which is referred to as 'co-simulation'. The `main notor control` module, shown as hatched, has its sampling frequency locally set to 15 kHz, whereas the 'system' frequency (set in the 'system properties' dialogue box) is set to 20 MHz. The latter is much higher, given the need to accurately represent 15 kHz voltage switching waveforms generated by the PWM unit of the converter. The content of the `main motor control` module, given in figure 3.12 shows the `voltage controller`, as discussed in subchapters 3.1.1.1, 3.1.2.2. However, the current and voltage measurement now makes use of the half-bridge inputs, as present in the hardware. In this case the current shunts in

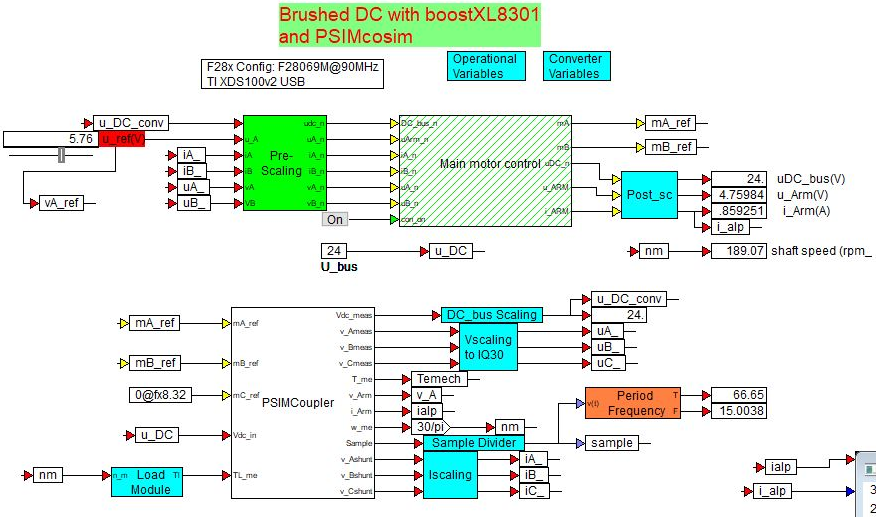


Fig. 3.11 PSIMcosim based drive, brushed DC operation, voltage control

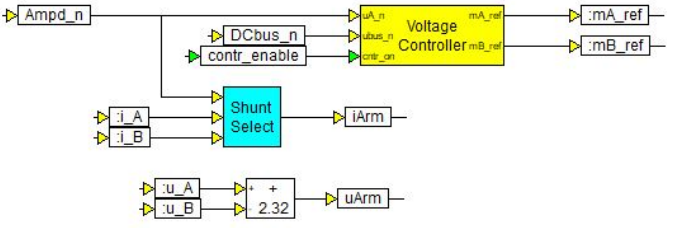


Fig. 3.12 Main motor control unit, brushed DC operation, voltage control

the lower converter switches, as shown in figure 3.13, are used to measure the current in two half-bridges used by the brushed DC machine. After sampling and signal-processing in the converter, these variables appear as:`::i_A`,`:i_B` in figure 3.12. A `Shunt select` module selects the signal that represents the measured, per unit armature current `IArm`. The low pass filtered voltages (which removes the PWM component) of the two half-bridges is also present in the `main notor control` module and represented by per unit variables:`::u_A`, `:u_B` respectively. The mathematical difference between these two signals is the per unit armature voltage `uArm`, which allows the user to check if this matches with the reference value.

Further observation of figure 3.13 shows the presence of the module `Brushed DC`, which is a generic model of the machine as presented earlier (see figure 2.11). However, the PSIM model, shown in figure 3.14, has an 'electrical interface' to allow use of the model with the converter.

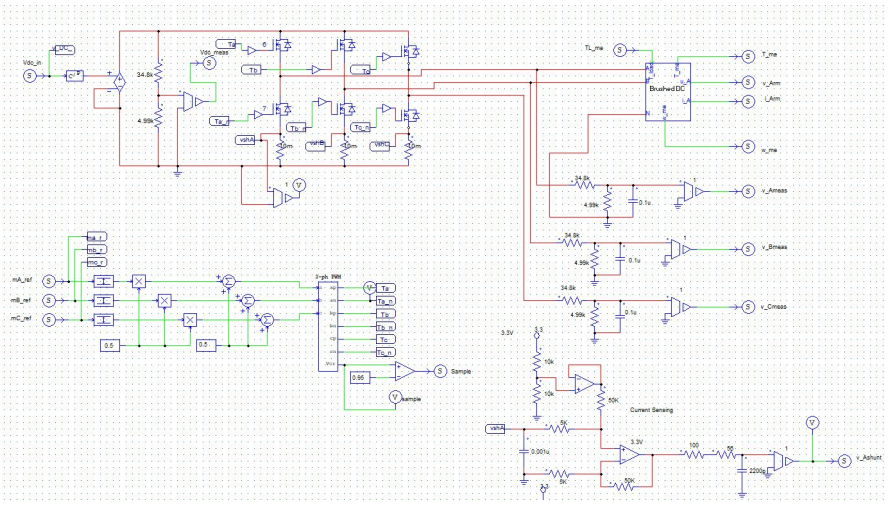


Fig. 3.13 Content of PSIM coupler module, brushed DC operation, voltage control

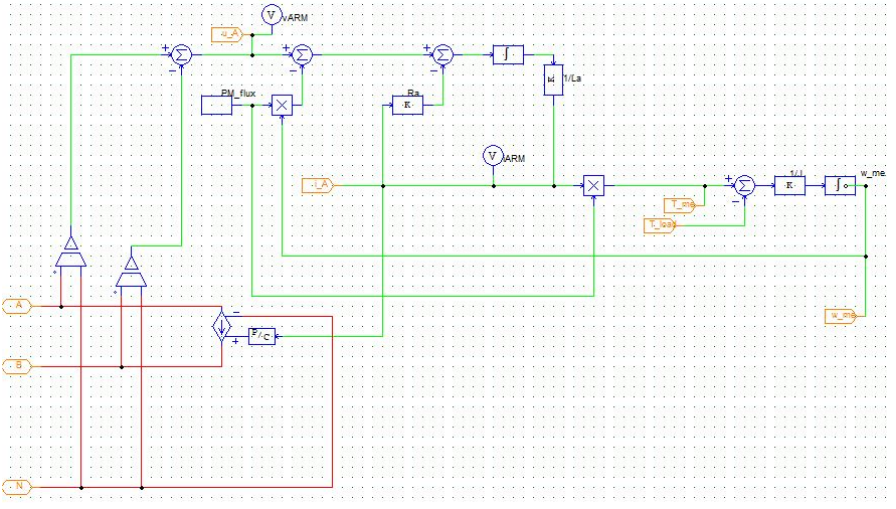


Fig. 3.14 PSIM sub-module: 'Brushed DC'

The converter used is represented by six ideal power electronic devices as shown in figure 3.13 and each of the three converter outputs is provided with a shunt resistor that measures the phase current in a half -bridge, provided that sampling is carried at the so called 'golden moment'(see figure 2.40). A PWM unit referred to as `3-ph PWM`, generates the gate control signals used to switch on/off the upper or lower devices of each converter leg. Connected to the output of each converter terminal is a resistance/capacitor network that is used to measure and low-pass filter the converter output voltage in

order to eliminate the PWM switching effects, as mentioned earlier. The three measured/ low pass filtered converter voltages and measured DC busvoltage are outputs of the `PSIMCoupler` module.

Central to sensorless operation of AC drives is the ability to accurately measure the phase voltages and currents. The experimental drive board(s) used in this GUIDE (and for this brushed DC drive) are mostly 'sensorless prepared', which implies that they have been purposely designed with a data acquisition module (ADC) that undertakes the signal conditioning of the voltage/currents as needed to realize sensorless (no mechanical encoder attached) control. In addition, access to voltages and currents, gives the user the ability to estimate motor input power, which in turn provides an indication of the DC bus current. The latter therefore, provides an estimate for the energy usage e.g. *Ah* of battery operated drives.

The converter model discussed here is 'sensorless prepared' and at this stage it is helpful to examine the data signal condition of the circuit in more detail, where use is made of figure 3.15. This is required because a number of critical hardware based parameters must be provided to the controller by the user, to ensure correct operation of the drive. Readily apparent in figure 3.15, is the

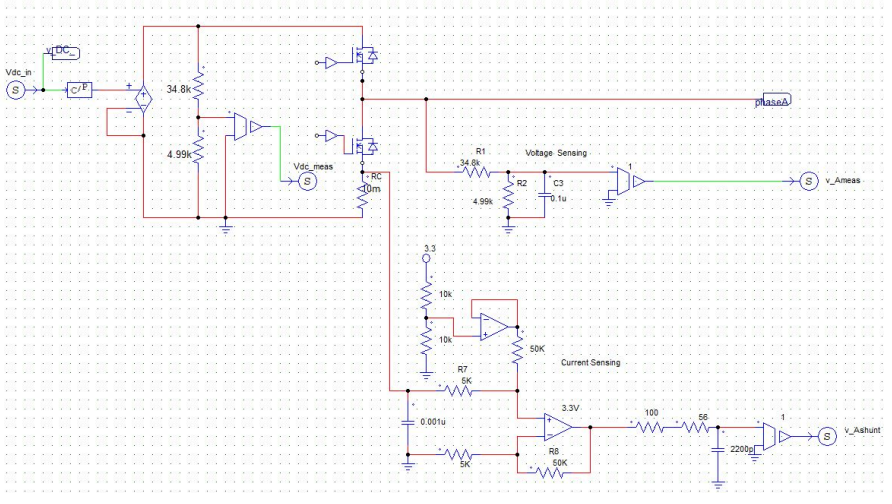


Fig. 3.15 One (of three) phase PSIM model of the BOOSTXL8301 unit

resistor/capacitor network used to attenuate and low-pass filter the converter output voltage to be used by the controller. The attenuation/gain G_R and corner frequency ω_c of the low pass filter circuit are equal to;

$$\mathrm{G_R} = \frac{R_2}{R_1 + R_2} \tag{3.2a}$$

$$\omega_\mathrm{c} = \frac{R_1 + R_2}{R_1\, R_2\, C_3} \tag{3.2b}$$

The corner frequency ω_c (rad/s) and attenuation of this filter are different on diferent boards, hence their value must be identified by inspection of the components (or provided via the circuit diagram). The example given in figure 3.15 shows the voltage filter components as present in the `BOOSTXL8301` module, which provide an attenuation of $G_R = 0.125$ and $\omega_c = 2291.3$ rad/s respectively. The low-pass filtered converter voltage(s) `v_Ameas` are sampled by the `Vscaling to IQ30` module shown in figure 3.12. When dealing with the actual MCU, the ADC converter takes over this sampling process. Measurement of the converter DC bus voltage is similar to that undertaken for the converter voltages.

An important part of the data acquisition process is measurement of the phase current using a shunt resistor `Rc`, in the converter (see figure 3.15). Furthermore, the current shunt voltage is amplified by an op-amp with gain G_sh, hence the voltage output `v_Ashunt` for the MCU (and an output of the PSIM module) is equal to;

$$u_\mathrm{sh} = i_\mathrm{conv}\, R_\mathrm{RC}\, G_\mathrm{sh} \tag{3.3}$$

The 'Converter Variables' dialog box (see figure 3.11), defines the scaling used by the Analog to Digital converter, which has a maximum input value of 3.3 V [14] and is directly linked to the signal processing undertaken for voltages and currents as discussed above. For example, the 'ADC-fullscale-voltage' value is the theoretical peak-to peak input voltage which must be applied to the low-pass filter network (see figure 3.15) to ensure that the ADC input level varies between $0 \rightarrow 3.3$V. Subsequent use of equations (3.2a),(3.3) shows that these values are defined as

$$\text{ADC} - \text{fullscalevoltage} = \frac{3.3}{G_{\text{R}}} \tag{3.4a}$$

$$\text{ADC} - \text{fullscalecurrent} = \frac{3.3}{R_{\text{RC}}\, G_{\text{sh}}} \tag{3.4b}$$

where G_{R}, is the low-pass filter attenuation used for measuring the converter voltages.

Similarly, the 'ADC-fullscale-current' value is the peak to peak current in the shunt resistance, which corresponds to an ADC input voltage range of $0 \rightarrow 3.3\,\text{V}$. Hence its value is determined by the shunt resistance value R_{RC} and the signal amplification gain G_{sh}, between the resistor and ADC input. The voltage and current measurements are unipolar and bipolar respectively. This implies that the voltage measurement is referenced to the negative converter DC bus voltage, whilst the current is referenced to the center of the ADC voltage span, i.e $^{3.3}/_{2}\,\text{V}$, as provided by an op-amp. Note that the ADC dialog values, are board dependent and need to be found by inspection of the circuit (or provided). The shunt voltage(s) `v_Ashunt`; are sampled by the `Iscaling` module shown in figure 3.11.
When dealing with the actual MCU, the ADC converter also takes over this sampling process. Apart from the measured voltage/currents generated by the converter, the PSIM coupler unit also provides the actual motor model armature current, voltage and speed, which are presented by the variables `v_A`, `ialp` and `nm` (after conversion to rpm) respectively (see figure 3.11).

Results achieved with the COSIM approach using the same armature voltage reference and simulation time as deployed for the SIM model (see figure 3.7) are shown in figure 3.16. Indicated in figure 3.16 is the armature voltage waveform due to the converter, together with the reference voltage. A zoom in around the 0.2 s time mark is also added to said figure to show the switching process and the impact on the armature current, e.g presence of a ripple component due to PWM. Co-simulation, as discussed in this sub-section, is instructive for understanding the impact of using a switching converter on drive operation.

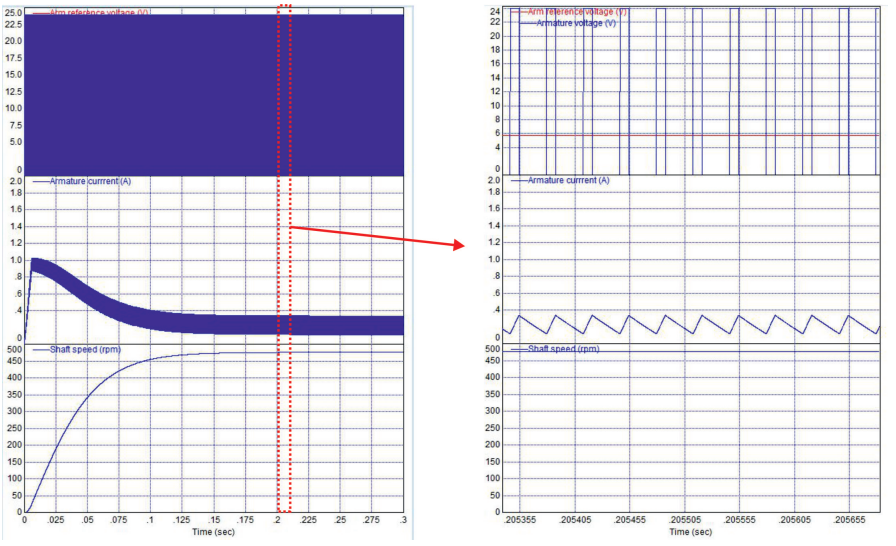


Fig. 3.16 Results COSIM, voltage control brushed DC

3.1.1.4 Voltage control-F069-boostXL8301: HIL

Reference files:

- `BrushDC-VControl-F28069M-BoostXL-DRV8301_HIL-compile.vsm`, for HIL-compile
- `BrushDC-VControl-F28069M-BoostXL-DRV8301_HIL-debug.vsm`,for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), as shown in figure 3.17, represents the `BOOSTXL-8301` converter (located on the `LAUNCHXL-F28069` module) which is connected to a brushed DC motor[25]. In the previous subsections, dealing with the SIM, PIL and COSIM toolchain

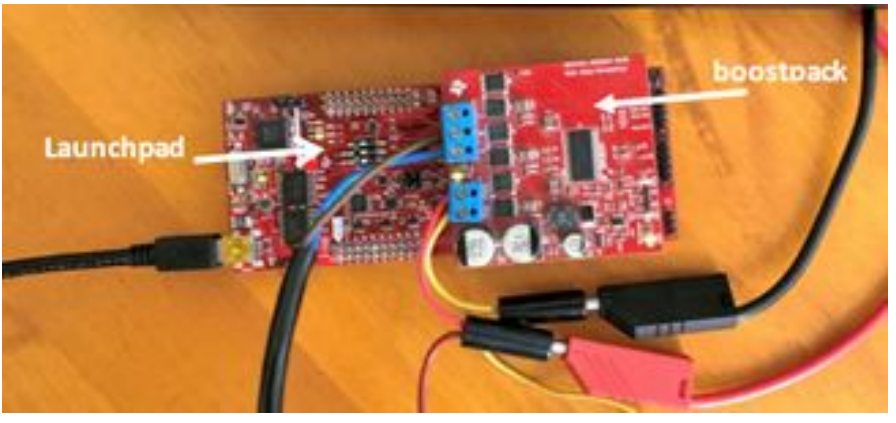


Fig. 3.17 MCU-converter setup for HIL operation

components, only limited knowledge of the control card and converter was required. For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate the file: `XXX.out` that is required in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation, with a given controller configuration.
The model given in figure 3.18 shows the 'voltage controller' module, which must be compiled to generate an out-file . Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user reference voltage `uA`, which is converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be

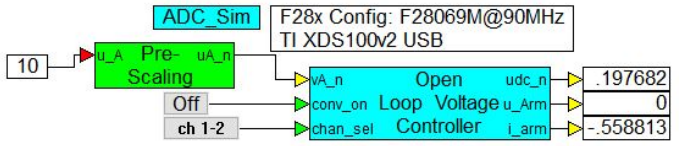


Fig. 3.18 HIL-Compile model of a brushed DC drive with voltage control

viewed. Further information on this tool will be given at a later stage. Outputs of the Controller module are the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts. In addition, the per unit armature voltage `u_Arm` and armature current `i_Arm` outputs are provided which must also be scaled in the HIL-debug stage.

Also shown in figure 3.18 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Open Loop Voltage Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialog boxes shown in figure 3.19.

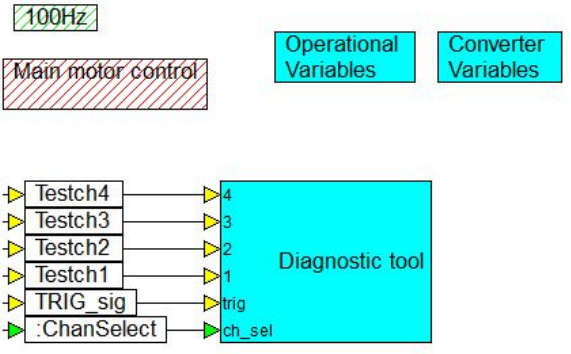


Fig. 3.19 One level down into the 'Open loop Voltage Controller' module

In this case two dialogue boxes are used, as shown in figure 3.19, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational variables' module has already been introduced in previous sub-sections and its content is shown in figure 3.20 to facilitate readability.

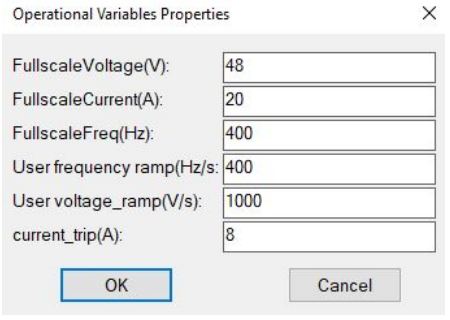


Fig. 3.20 HIL: Dialogue box 'Operational Variables'

The 'operational Variables' dialog box, is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage. The 'Converter Variables' dialog box, as given in figure 3.21, defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.4).

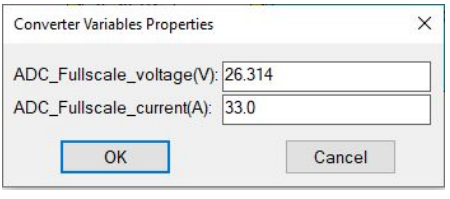


Fig. 3.21 HIL: Dialogue box 'Converter Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph when operating in HIL-debug. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the brushed DC drive. Among these is the ADC offset control task (a state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 3.19 is the `Main motor control` module which contains the main control algorithm as needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) signal generated by the ADC unit. Moving one

level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 3.22. Of these the 'Voltage controller' has been discussed previously, given that it is the module used in phases : SIM, PIL and COSIM of the development toolchain.

The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured for with the following basic settings :

- PWM frequency: 15 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: 0.11 μs
- Board in use: `LAUNCHXL-F28069` with `BOOSTXL-8301` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed. The converter dead-time for the PWM is set to 10 MCU clock cycles. Moving one level down the `ADC-PWM` unit, see figure 3.23

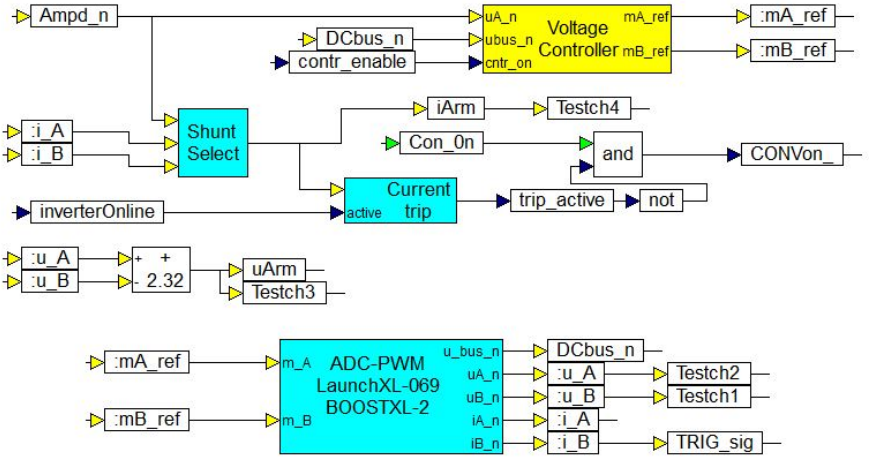


Fig. 3.22 One level down into the 'Main motor controller' module

shows the `PWM unit BOOSTXL-2` that controls the converter switches based on the modulation indices generated by the controller (see figure 3.23). The `ADC BOOSTXL-2` unit is used to measure the two low-pass filtered half-bridge voltages `uA_N`, `uB_n` and currents `iA_n`, `iB_n` respectively. ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `uDC_n`, completes the ADC

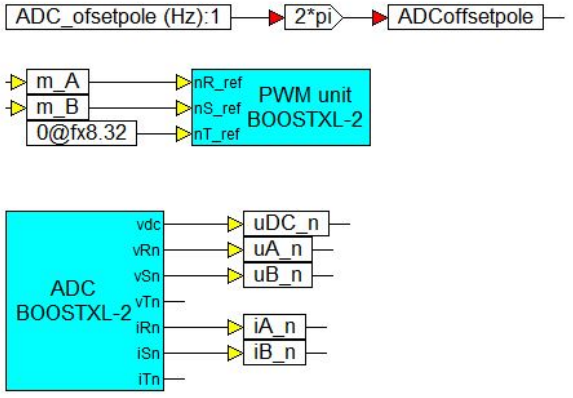


Fig. 3.23 One level down into the 'ADC-PWM' module

measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example, as shown in figure 3.24 and its task is to execute the `XXX.out` file compiled in the 'HIL-compile' phase (see above).

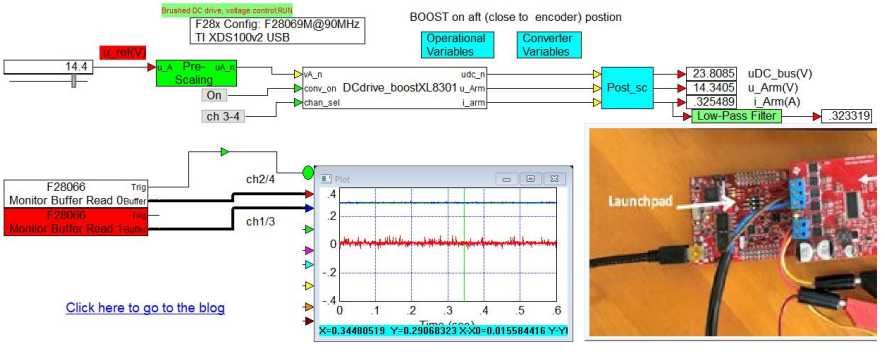


Fig. 3.24 HIL-Debug model of a brushed DC drive with voltage control

Central to this figure is the Embed 'target execution module', which executes said out-file.
A slider is used to set the armature voltage reference amplitude. Post-scaling modules are used to convert the per-unit measured DC bus voltage and armature voltage to actual voltages, as shown with the numeric displays. In addition, a post-scaling unit is present to convert the measured per unit

armature current to its actual value also shown on a numeric display. Two 'Monitor Buffer' modules are used to display two selected diagnostic signals (using a button connected to the `Ch_sel` input). The scope given in figure 3.24, displays the per unit armature current (in 'red') and the per unit armature voltage (in 'blue'). This requires the channel select button to be set to `ch 3-4`. Note that the voltage reference slider is set to `u_ref=14.4`V, hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{14.4}/_{48} = 0.3$, given that a full-scale voltage of u_{fs}=48 V is used. Observation of the scope result confirms that the amplitude is approximately equal to the anticipated value. Some amplitude deviation will inevitably occur due to practical issues such as: converter dead-time and voltage drop across the converter switches.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialog boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialog boxes entry specified under HIL-compile. The dialog boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialog box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive checklist, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope, which in this example (figure 3.24) is set to show the per unit variables u^n_{arm}, i^n_{arm} on the scope module.

For comparative purposes an oscilloscope with DC current/voltage probes was used to measure the armature current and the phase 'A' half-bridge voltage, as shown in figure 3.25. The armature current shown on the numeric display, should represent the current waveform shown on the oscilloscope (figure 3.25), as indeed it does. Note that the average voltage per sample voltage for the

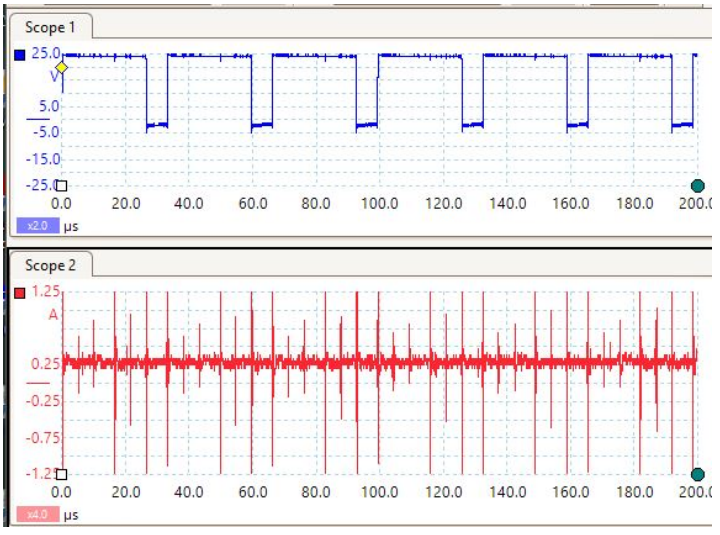


Fig. 3.25 experimental results: measured armature current 'red' and half-bridge converter voltage, phase 'A'

converter voltage, shown in figure 3.25 (upper sub-plot) must be equal to $\frac{u_{DC}}{2} + \frac{u_{ref}}{2}$. Correspondingly, for $u_{ref} = 14.4$V, the average voltage per sample (AVS) value must be $u_A = 19.2$V, as is indeed the case.

At this stage, the reader has passed through all of the development stages needed to implement a fully operational electrical drive. In future example the control structure and motor topology will change, but the development tool-chain will remain the same. Hence it is important for the reader to be familiar with all aspects of the material described in this section.

3.1.1.5 Voltage control-F0049-GANINV: HIL

Reference files:

- `V_F280049-GANINV-compile.vsm`, for HIL-compile
- `V_F280049-GANINV-run.vsm`,for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) as shown in figure 3.26 represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F280049` module), which is connected to a brushed DC motor[25]. For HIL operation the drive board itself needs to be known in considerable

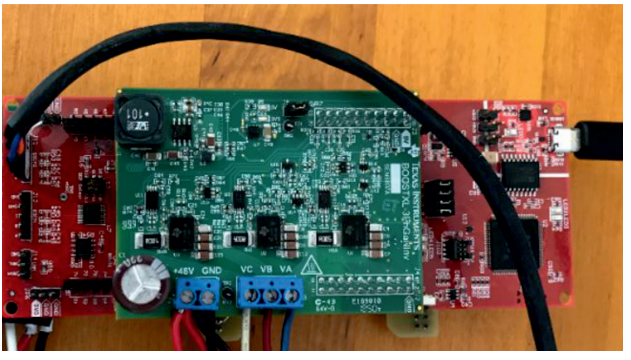


Fig. 3.26 MCU-converter setup for HIL operation

detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs, which activate the converter switches. A compile stage is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 3.27 shows the 'voltage controller' module, which must be compiled to generate an out-file.

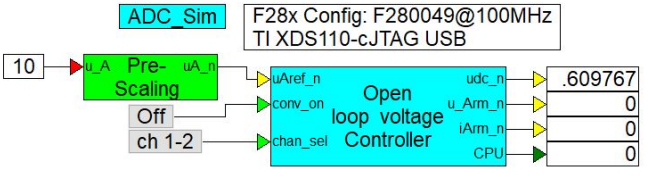


Fig. 3.27 HIL-Compile model of a brushed DC drive with voltage control

Inputs, shown in the stage as constants (for readability purposes) for this controller module include the user reference voltage `uA`, which is converted to 'per unit' values in the 'pre-scaling' module.
Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Further information on this tool will be given at a later stage.

Outputs of the Controller module are the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts. In addition, the per unit armature voltage `u_Arm` and armature current `i_Arm` outputs are provided which must also be scaled in the HIL-debug stage. Furthermore,a `CPU` output is added, which provides the user with a % reading of MCU activity, relative the controller sample time.

Also shown in figure 3.27 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialog boxes shown in figure 3.28. Two dialogue boxes are used, as shown in figure 3.28, which contain the param-

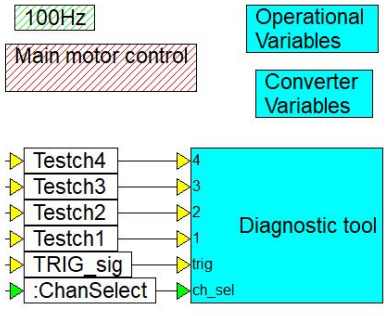


Fig. 3.28 One level down into the 'Open loop Voltage Controller' module

eters that need to be assigned by the user for HIL-operation. The 'operational Variables' dialog box (see figure 3.29), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage. The 'Converter Variables' dialogue box, as given in figure 3.30, defines the scaling used by the Analogue to Digital converter, as discussed earlier (see equation (4.4)).

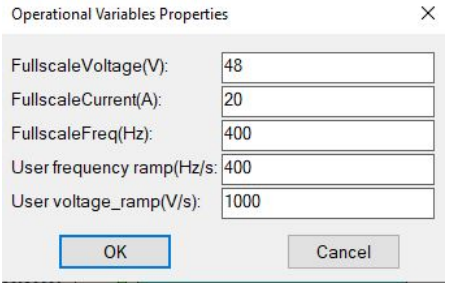


Fig. 3.29 HIL: Dialogue box 'Operational Variables'

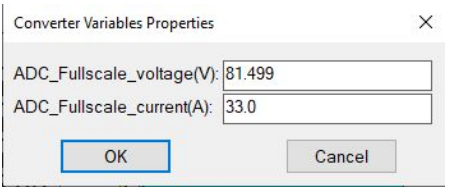


Fig. 3.30 HIL: Dialogue box 'Converter Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph when operating in HIL-debug. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 3.28 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 3.31. Of these, the 'Voltage controller' has been discussed previously, for a different MCU, see figure 3.6, where it was used in phases : SIM, PIL and COSIM of the respective development tool-chain.

The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 20 kHz

- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F280049` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

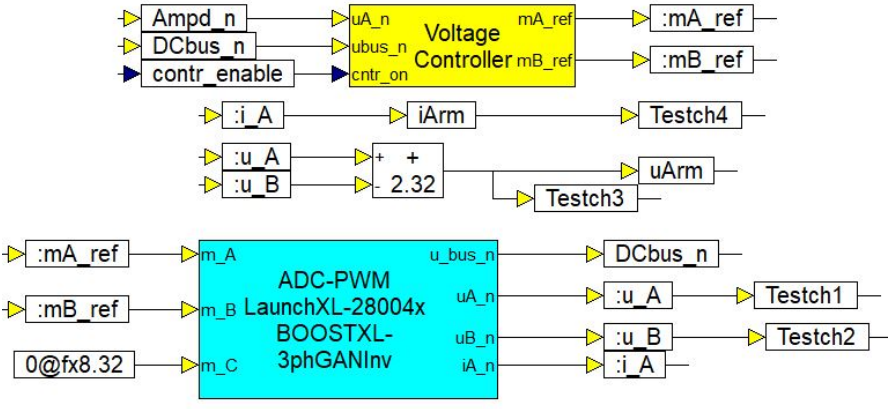


Fig. 3.31 One level down into the 'Main motor controller' module

Note that the `GANINV` converter board makes use of in-line phase current transducers, hence for this application phase 'A' unit is used for armature current measurement `i_A`.
Moving one level down the `ADC-PWM` unit, see figure 3.32 shows the `PWM unit BOOSTXL-GANINV` that controls the converter switches, based on the modulation indices generated by the controller (see figure 3.31).
The `ADC BOOSTXL 3PH-GANINV` unit is used to measure the two low-pass filtered half-bridge voltages `uA_n`, `uB_n` and current `iA_n` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offsets is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `uDC_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of the laboratory (see figure 3.33), and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. A slider is used to set the armature voltage reference amplitude .

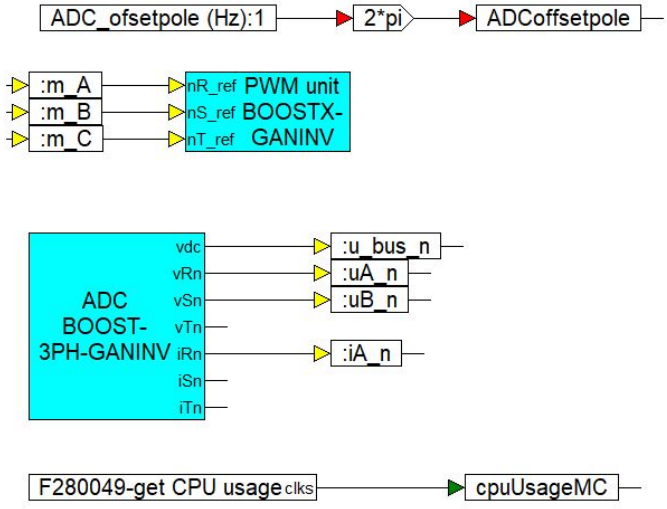


Fig. 3.32 One level down into the 'ADC-PWM' module

Post-scaling modules are used to convert the per-unit measured DC bus voltage and armature voltage to actual voltages, as shown with numeric displays. In addition, a post-scaling unit is present to convert the measured per unit armature current to its actual value, also shown on a numeric display. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

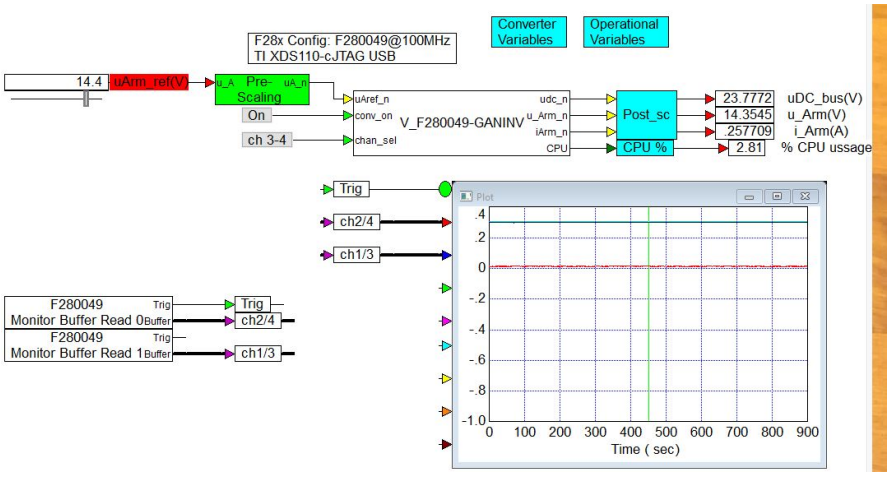


Fig. 3.33 HIL-Debug model, F0049-GANINV of a brushed DC drive with voltage control

The plot given in figure 3.33 displays the per unit armature current (in 'red') and the per unit armature voltage. This requires the channel select button to be set to `ch 3-4`. Note that the voltage reference slider is set to `u_ref=14.4`V, hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{14.4}/_{48} = 0.3$, given that a full-scale voltage of u_{fs}=48 V is used. Observation of the plot result confirms that the amplitude is approximately equal to the anticipated value. Some amplitude deviation will inevitably occur due to practical issues such as: converter dead-time and voltage drop across the converter switches. The measured armature current in the numerical display shows a value of 0.25 A, which corresponds to a per unit value of ${}^{0.25}/_{20} = 0.0125$, given a fullscale current value of $i_{fs} = 20$(A).

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive checklist, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope, which in this example (figure 3.33) is set to show the per unit variables u^n_{arm}, i^n_{arm} on the scope module.

For comparative purposes an oscilloscope with DC current/voltage probes was used to measure the armature current and the phase 'A' half-bridge voltage, as shown in figure 3.34.
The armature current shown on the numeric display, should represent the current waveform shown on the oscilloscope (figure 3.34), as indeed it does.

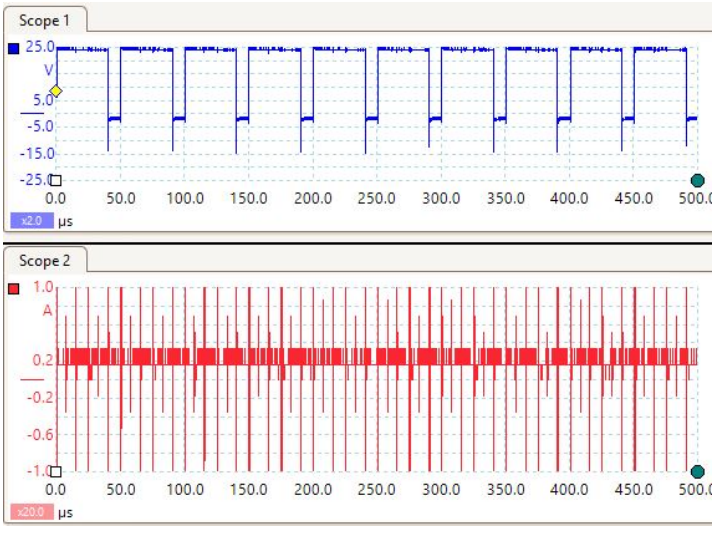


Fig. 3.34 Experimental results: measured armature current 'red' and half-bridge converter voltage 'blue', phase 'A'

Note that the average per sample (AVS) voltage for the converter voltage, shown in figure 3.34 (upper sub-plot) must be equal to $\frac{u_{DC}}{2} + \frac{u_{ref}}{2}$. Correspondingly, for $u_{ref} = 14.4$V, the average voltage per sample (AVS) value must be $u_A = 19.2$V, as is indeed the case.

3.1.2 Armature Voltage Control-STM

Purpose of this section is to realize open loop armature voltage control of a brushed DC machine using an ST-Micro board. At the same time this type of example is used to make a functional check on the current/voltage measurements and overall functioning of the drive board and software. The tool chain as introduced in figure 2.45 is used as a template for the following subsections.

3.1.2.1 Voltage control-F303-ESC001: SIM

Reference file: `DC_STM_V_SIM.vsm`

This example considers the simulation of a brushed DC motor [25] connected to a ST-Micro `ESC001` module, as shown in figure 3.35. The aim is to examine

operation of the drive using a fixed point controller, that will subsequently be used in the HIL implementation. The motor is connected to phases 'A' and 'B', hence 'C' is not used in the example.

Drilling into the 'voltage controller' reveals the controller structure shown

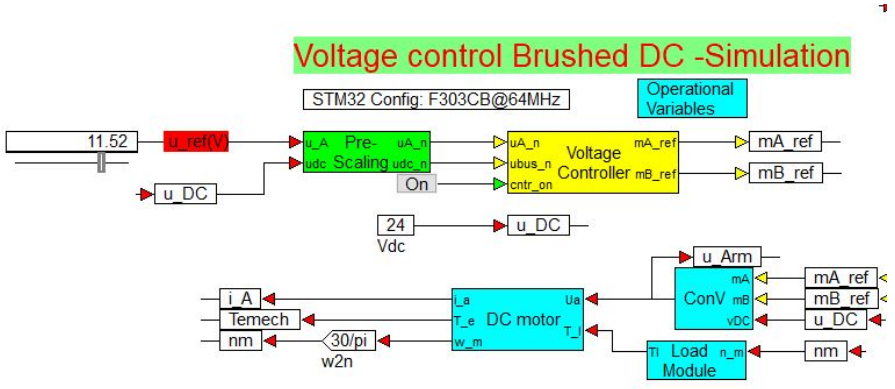


Fig. 3.35 Brushed DC drive with armature voltage control

in figure 3.36. Outputs of the controller are the two modulation indices that controller the two converter half bridges in use. For all controllers consid-

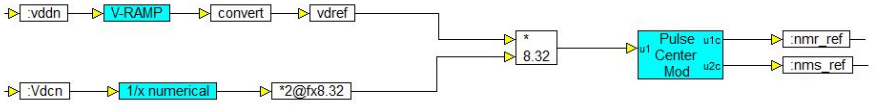


Fig. 3.36 Voltage controller structure

ered, the aim is to generate the modulation indices that are required by the PWM units. Consequently, it is important to convert the reference voltage to modulation index using experession (3.5).

$$m^{\mathrm{ref}} = \frac{2\,u^{\mathrm{ref}}}{u_{\mathrm{DC}}} \tag{3.5}$$

which requires access to the measured DC bus voltage u_{DC}, shown as u_{DC}. The 'Pulse Center Mod' unit splits the reference modulation index into components $\pm \frac{m^{\mathrm{ref}}}{2}$ as required to modulate the two half-bridges used. The voltage

supplied to the motor `U_Arm` is found by multiplying the modulation indices `mA_ref`, `mB_ref` by half the DC bus voltage as shown in figure 3.35 and taking the difference between the two half bridge voltages (using module '`ConV`'). These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables $u_a^{\text{ref}}(t_k)$, $u_b^{\text{ref}}(t_k)$.
The aim of a switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sub chapter 2.1.5). In this case the armature is directly connected between two converter phases and this implies that the maximum modulation index is equal to $m_{max} = 2$, which corresponds to an armature voltage of 24V. A quadratic load is introduced, which provides a value of 100mNm at a shaft speed of 1000rpm.

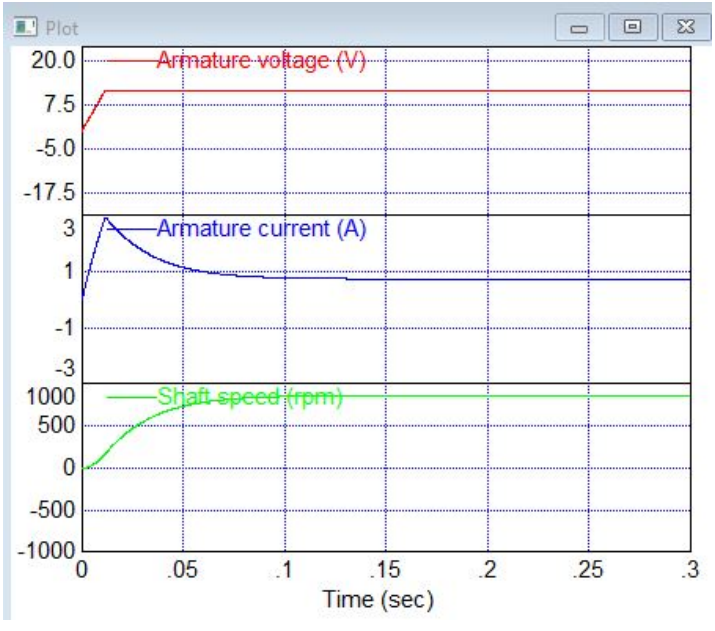


Fig. 3.37 SIM result, brushed DC operation, voltage control

The results shown in figure 3.37, confirm the presence of armature voltage control. Note that at start up a ramp function is used, which determines the rate of reference variable change, as set in the 'operational Variables' dialogue box. The steady state shaft speed is governed by the armature voltage setting and the applied load characteristic.

3.1.2.2 Voltage control-F303-ESC001: PIL

Reference files:

- `BrushDC-VControl-F303-ESC001-PIL.vsm`
- `DC_Vctrl_STM_PIL.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F303. This implies that the controller module shown in the simulation (see figure 3.36) is now replaced with a file: `DC_Vctrl_STM_PIL` as indicated in figure 3.38. The target interface model that executes the out-file, gen-

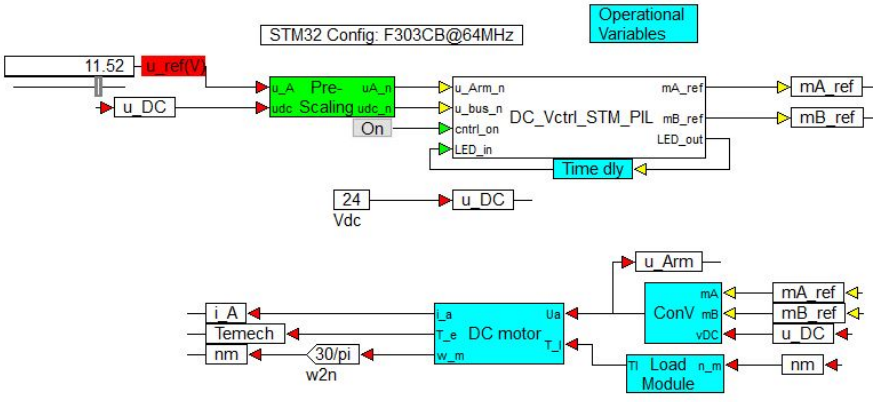


Fig. 3.38 PIL based drive, brushed DC operation, voltage control

erates the modulation indices required for the converter motor and these components are not affected by the use of the PIL. Generation of the out file is based on the PIL module, see figure 3.39, which must be compiled using the command: `toolbox>codegen`. Moving one level into the PIL module

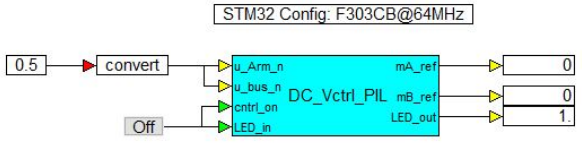


Fig. 3.39 PIL module, brushed DC operation, voltage control

`DC_Vctrl_PIL` shows the voltage controller as introduced in section 3.1.2.1.

Added to the voltage controller is a module `LED_ctrl`, which is used to flash

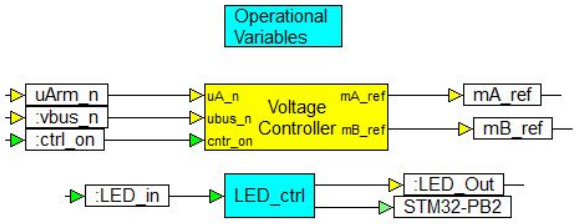


Fig. 3.40 Content PIL module, brushed DC operation, voltage control

an LED on the MCU board to show that the PIL control is operational.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational. Results produced by PIL operation are identical to the plot shown in figure 3.37 and are therefore not shown here.

3.1.2.3 Voltage control-F303-ESC001:COSIM

Reference files:

- `BrushDC-VControl-F303-ESC001-COSIM.vsm`
- `ESC001_DC_mot.psimsch`
- `DC motor model_v1.psimsch`

To evaluate operation of a controller with a switching converter, in this case the `STM-ESC001`, use can be made of a `PSIMCoupler` module, as indicated in figure 3.41. This approach implies the use of PSIM together with Embed, which is referred to as 'co-simulation'. The `main notor control` module , hatched in this case as its sampling frequency is locally set to 10 kHz, whereas the 'system' frequency (set in the 'system properties' dialogue box) is set to 20 MHz. The latter is much higher, given the need to accurately represent 10 kHz voltage switching waveforms generated by the PWM unit of the converter. The content of the `main motor control` module, given in figure 3.42 shows the `voltage controller`, as discussed in subchapters 3.1.2.2, 3.1.1.1. However, the current and voltage measurement now makes use of the half-bridge inputs, as present in the hardware. In this case the current shunts in the lower converter switches, as shown in figure 3.43, are used to generate the current in two half-bridges used by the brushed DC machine. After sampling and signal-processing in the converter, these variables appear as:`::i_A`,`:i_B` in figure 3.42. A `Shunt select` module selects the signal that represents the

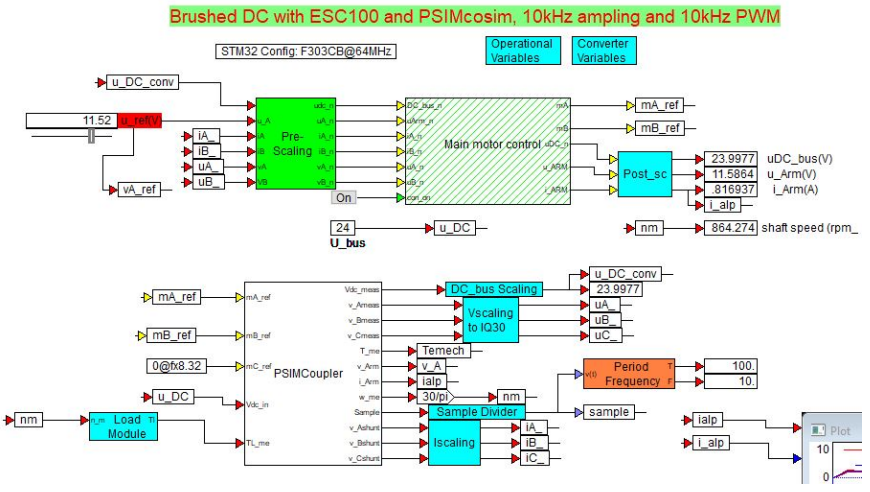


Fig. 3.41 PSIMcosim based drive, brushed DC operation, voltage control

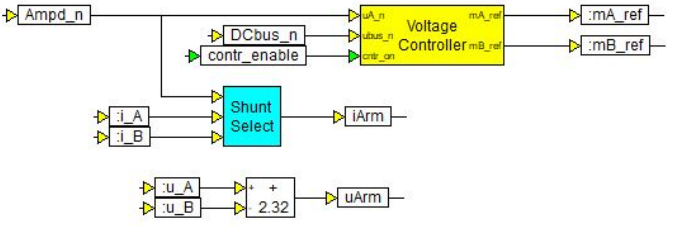


Fig. 3.42 Main motor control unit, brushed DC operation, voltage control

measured, per unit armature current `iArm`. The low pass filtered voltages (which removes the PWM component) of the two half-bridges is also present in the `main notor control` module and represented by per unit variables:`:u_A`, `:u_B` respectively. The difference between these two signals is in fact the per unit armature voltage `uArm`, which allows the user to check if this matches with the reference value.

Note that in this figure only one of the three `current sensing` units is shown, which does the signal processing of the measured shunt current. Further observation of figure 3.43 shows the presence of a module `Brushed DC`, which is a generic model of the machine as presented earlier (see figure 2.11). However, the PSIM model, also shown earlier, see figure 3.14, has an 'electrical interface' to allow use of said model with the converter. The converter used is represented by six ideal power electronic devices as shown in figure 3.43 and each of the three converter outputs is provided with a shunt resistor that measures the phase current in a half -bridge, provided that sampling is carried at the so called 'golden moment'(see figure 2.40). A PWM unit referred to as `3-ph PWM`,

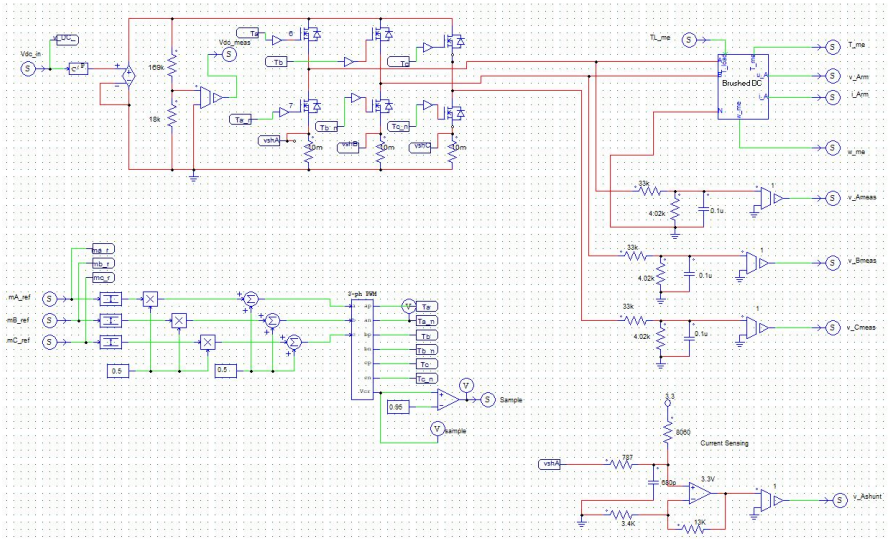


Fig. 3.43 Content PSIM coupler module, brushed DC operation, voltage control

generates the gate control signals used to switch on/off the upper or lower devices of each converter leg. Connected to the output of each converter terminal is a resistance/capacitor network that is used to measure and low-pass filter the converter output voltage in order to eliminate the PWM switching effects, as mentioned earlier. The three measured/ low pass filtered converter voltages and measured DC busvoltage are outputs of the `PSIMCoupler` module.

Central to sensorless operation is the ability to accurately measure the phase voltages and currents. The experimental drive board(s) used in this GUIDE are mostly 'sensorless prepared', which implies that they have been purposely designed with a data acquisition module (ADC) that undertakes the signal conditioning of the voltage/currents as needed to realize sensorless (no mechanical encoder attached) control. In addition, access to voltages and currents, gives the user ability to estimate the motor input power, which in turn provides an indication of the DC bus current. The latter therefore, provides an estimate for the energy usage e.q. Ah of battery operated drives.

The 'Converter Variables' dialog box, as given in figure 3.41, defines the scaling used by the Analog to Digital converter, which has a maximum input value of 3.3 V and is directly linked to the signal processing undertaken for voltages and currents as discussed above, as discussed earlier. Note that the ADC dialog values, are board dependent and need to be found by inspection of the circuit (or provided). The shunt voltage(s) `v_Ashunt`; are sampled by the `Iscaling` module shown in figure 3.41. When dealing with the actual MCU, the ADC converter also takes over this sampling process. Apart from the

measured voltage/currents generated by the converter, the PSIM coupler unit also provides the actual motor model armature current, voltage and speed, which are presented by the variables `v_A`, `ialp` and `n_m` (after conversion to rpm) respectively, see figure 3.41.

Results achieved with the COSIM approach using the same armature voltage reference and simulation time as deployed for the SIM model (see figure 3.37) are shown in figure 3.44, for a zoomed in time frame around the 0.124s time mark, in order to show the converter switching process. Indicated in

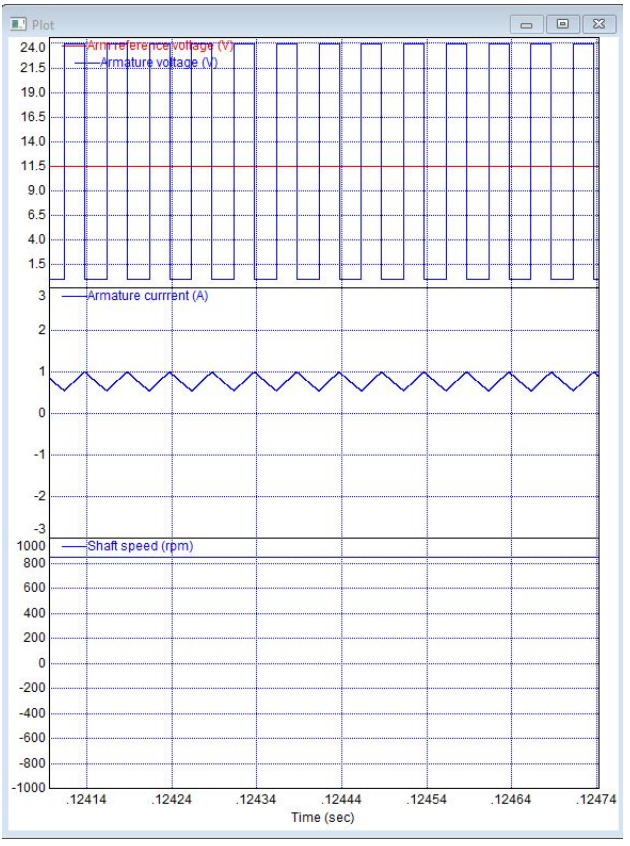


Fig. 3.44 result COSIM, voltage control brushed DC

figure 3.44 is the armature voltage waveform due to the converter together with the reference voltage. Co-simulation, as discussed in this sub-section, is instructive for understanding the impact of using a switching converter on drive operation.

3.1.2.4 Voltage control-F303-ESC001: HIL

Reference files:

- `DC_F303RE_ESC001_compile-HIL.vsm`, for HIL-compile
- `DC_F303RE_ESC001_run-HIL.vsm`,for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), shown in figure 3.45 represents the `STM-ESC001` module, which is connected to a brushed DC motor[25] In the previous subsections, dealing with the SIM, PIL and

Fig. 3.45 MCU-converter setup for HIL operation

COSIM tool-chain components, only limited knowledge of the control card and converter was required. For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 3.46 shows the 'voltage controller' module, which must be compiled to generate an out-file. Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user reference voltage `u_A`, which is converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Further information on this tool will be given at a later stage. Outputs of the Controller module are the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts. In addition, the per unit

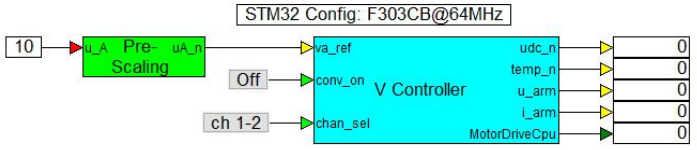


Fig. 3.46 HIL-Compile model of a brushed DC drive with voltage control

armature voltage `u_Arm`, temperature and armature current `i_Arm` outputs are provided which must also be scaled in the HIL-debug stage.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialog boxes shown in figure 3.47. In this case two dialogue boxes are used, as shown in figure 3.47, which con-

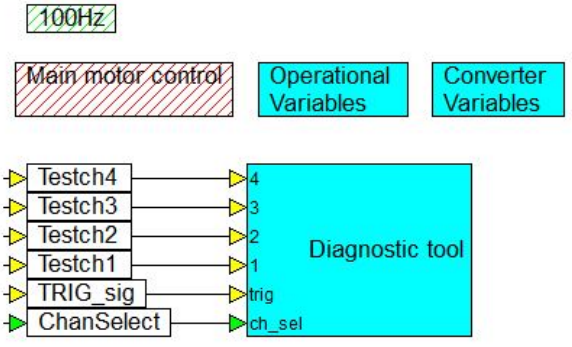


Fig. 3.47 One level down into the 'Open loop Voltage Controller' module

tain the parameters that need to be assigned by the user for HIL-operation. The 'operational variable' module has already been introduced in previous sub-sections and its content is shown in figure 3.48 to facilitate readability. The 'operational Variables' dialogue box, is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage. The 'Converter Variables' box, as given in figure 3.49, defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation 4.4. A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph when operating in HIL-debug. A 'two channel' multiplexer is used to allow the user to either display variables `Test1`, `Test2` or `Test3`, `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among

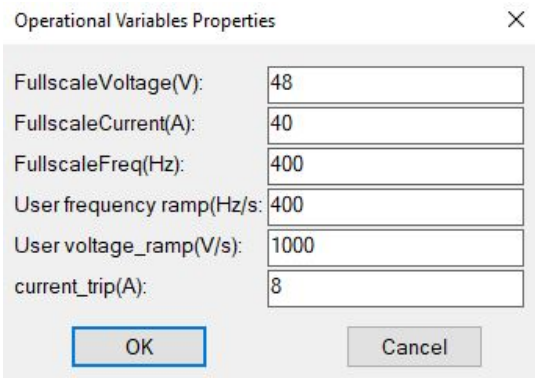


Fig. 3.48 HIL: Dialogue box 'Operational Variables'

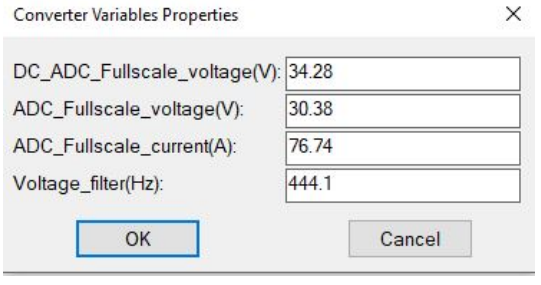


Fig. 3.49 HIL: Dialogue box 'Converter Variables'

these is the ADC offset control task (State machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 3.47 is the `Main motor control` module which contains the main control algorithm as needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 3.50. Of these the 'Voltage controller' has been discussed previously, given that it is the module used in phases : SIM, PIL and COSIM of the development toolchain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured for with the following basic settings :

- PWM frequency: 10 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs

- Board in use: `STM-F303` module converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed, this implies triggering `DMA1_Channel1.0.1`. Moving

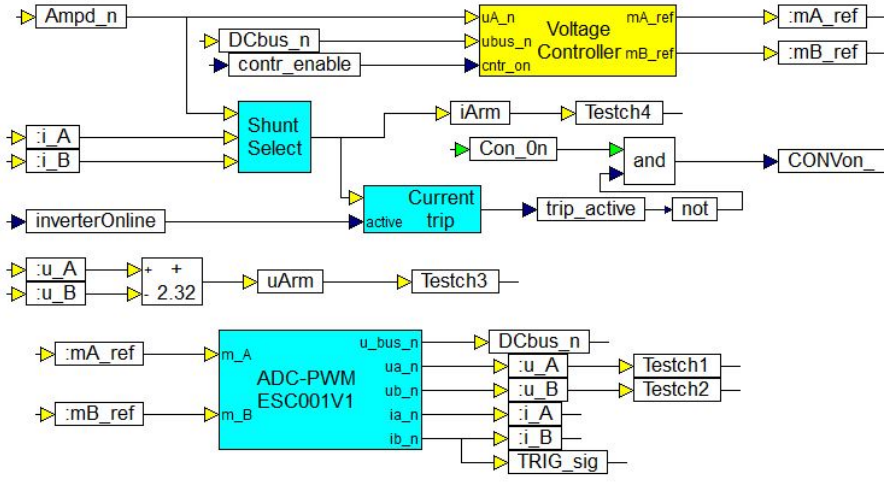


Fig. 3.50 One level down into the 'Main motor controller' module

one level down the `ADC-PWM` unit (see figure 3.51), shows the `PWM unit` that controls the converter switches based on the modulation indices generated by the controller. The `ADC` unit is used to measure the two low-pass filtered half-bridge voltages `uA_N`, `uB_n` currents `iA_n`, `iB_n` and temperature respectively. ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialog box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `ubus_n`, completes the ADC measurement sequence. Also added to said module is a '`STM32-get-CPU usage` module which is used to show the % CPU activity relative the sampling time.

The HIL-debug/run tool-chain component is the operational component of the laboratory, shown in figure 3.52 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. A slider is used to set the armature voltage reference amplitude. Post-scaling modules are used to convert the per-unit measured DC bus voltage and armature voltage to actual voltages, as shown with numeric

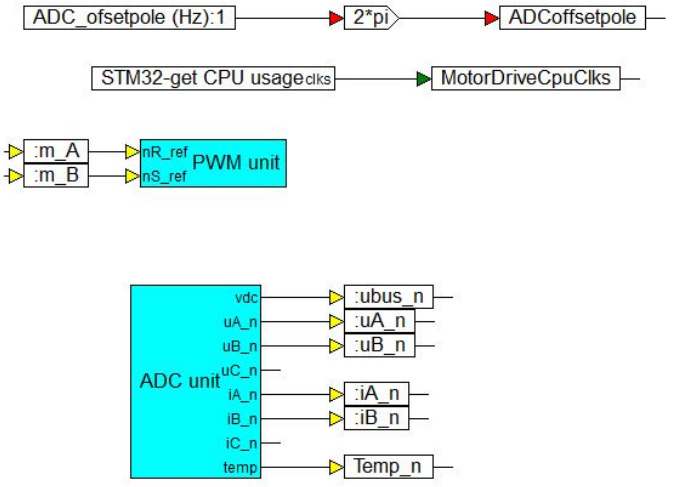


Fig. 3.51 One level down into the 'ADC-PWM' module

displays. In addition, the converter temperature and % CPU time, relative to the sampling time is shown. In addition, a post-scaling unit is present to convert the measured per unit armature current to its actual value also shown on a numeric display. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

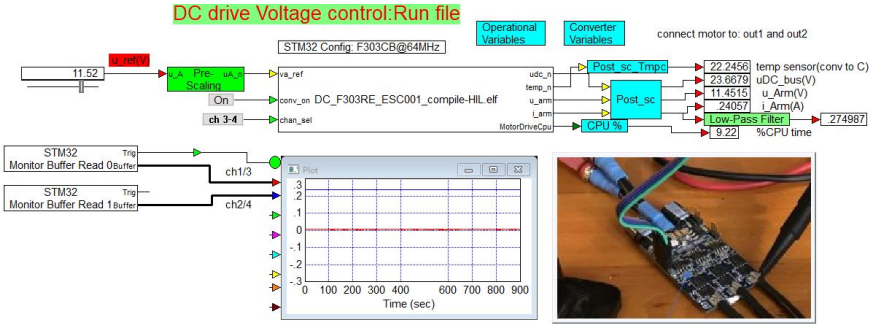


Fig. 3.52 HIL-Debug model of a brushed DC drive with voltage control

The plot given in figure 3.52 displays the per unit armature current (in 'red') and the per unit armature voltage. This requires the channel select button to be set to `ch 3-4`. Note that the voltage reference slider is set to `u_ref=11.52`V, hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{11.52}/_{48} = 0.24$, given that a full-scale voltage of u_{fs}=48 V is used. Observation of the plot result confirms that the amplitude is approximately equal to the anticipated value.

Some amplitude deviation will inevitably occur due to practical issues such as: converter dead-time and voltage drop across the converter switches.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialog boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialog boxes entry specified under HIL-compile. The dialog boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive checklist, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope, which in this example (figure 3.52) is set to show the per unit variables u^n_{arm}, i^n_{arm} on the plot module.

For comparative purposes an Oscilloscope with DC current/voltage probes was used to measure the armature current and the phase 'A' half-bridge voltage, as shown in figure 3.53.
The armature current shown on the numeric display, should be conform with the current waveform shown on the oscilloscope (figure 3.53), as indeed it does. Note that the average voltage per sample (AVS) voltage for the converter voltage, shown in figure 3.53 (upper sub-plot) must be equal to $\frac{u_{DC}}{2} + \frac{u_{ref}}{2}$. Correspondingly, for $u_{ref} = 11.52$V, the average voltage per sample (AVS) value must be $u_A = 17.7$V, as is indeed the case. The armature current, measured using a current probe is equal to approximately 0.2 A.

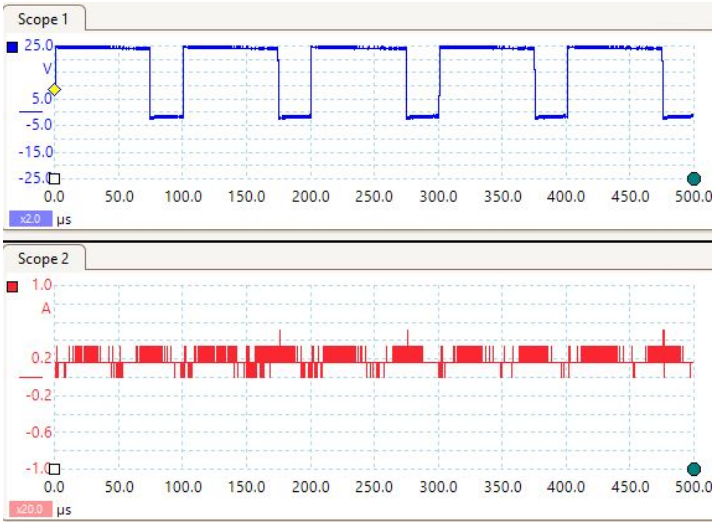


Fig. 3.53 Experimental results: measured armature current 'red' and half-bridge converter voltage, phase 'A'

3.2 Armature Current Control

In this section, so called 'model based current control' as introduced in section 2.1.4 is applied to a set of examples for brushed DC machines using Texas Instruments and ST-Micro hardware. Central to this approach is knowledge of the armature resistance, inductance and control sampling frequency. These variables define the current controller proportional gain and bandwidth, see equation (2.32).

3.2.1 Armature Current Control-TI

Purpose of this section is to realize armature current control of a brushed DC machine using Texas Instruments hardware. The tool chain as introduced in figure 2.45 is again used as a template for the ensuing subsections.

3.2.1.1 Current control-F069-boostXL8301: SIM

Reference file: `BrushDC-CControl-F28069M-BoostXL-DRV8301-SIM.vsm`

This example considers the simulation of a brushed DC motor [25] connected to a Texas Instruments F28069M Launchpad, with a three-phase

boostXL-DRV8301 converter, as shown in figure 3.54. The aim is to examine operation of the drive using a fixed point controller, that will subsequently be used in the HIL implementation. The motor is connected to phases 'A' and 'B', hence 'C' is not used in the example.

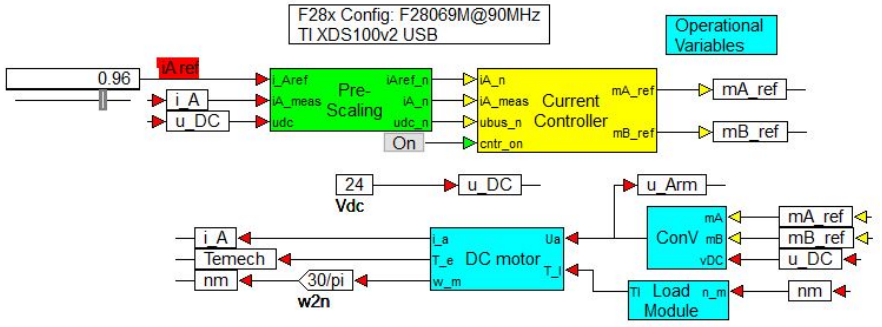


Fig. 3.54 Brushed DC drive with armature current control

Drilling into the 'Current controller' reveals the controller structure shown in figure 3.55. Outputs of the controller are the two modulation indices that controller the two converter half bridges in use. For all controllers consid-

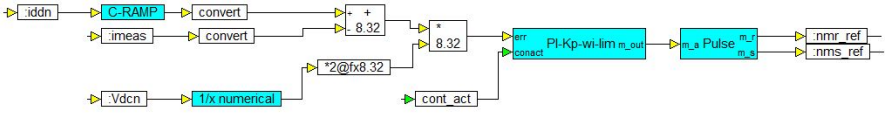


Fig. 3.55 Current controller structure

ered, the aim is to generate the modulation indices that are required by the PWM units. Consequently, it is important to convert the reference voltage to modulation index using expression (3.6).

$$m^{\text{ref}} = \frac{2\,u^{\text{ref}}}{u_{\text{DC}}} \tag{3.6}$$

which requires access to the measured DC bus voltage u_{DC}, shown as `Vdcn` (a per unit variable) in figure 3.55. Model based current control, as discussed in section 2.1.4 is applied in unit `PI-Kp-wi-Lim`. Correspondingly, the gain and

bandwidth of said unit are defined by the armature inductance, resistance and controller sampling frequency (see equation (2.32)). The 'Pulse' unit splits the reference modulation index into components $\pm \frac{m^{\text{ref}}}{2}$ as required to modulate the two half-bridges used. The voltage supplied to the motor `U_Arm` is found by multiplying the modulation indices `mA_ref`, `mB_ref` by half the DC bus voltage as shown in figure 3.54 and taking the difference between two half bridge voltages (use of module '`ConV`'). These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables $u_a^{\text{ref}}(t_k)$, $u_b^{\text{ref}}(t_k)$. The aim of a switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sub chapter 2.1.5). In this case the armature is directly connected between two converter phases and this implies that the maximum modulation index is equal to $m_{max} = 2$, which corresponds to an armature voltage of 24V. A quadratic load is introduced, that provides a value of 100mNm at a shaft speed of 1000rpm.

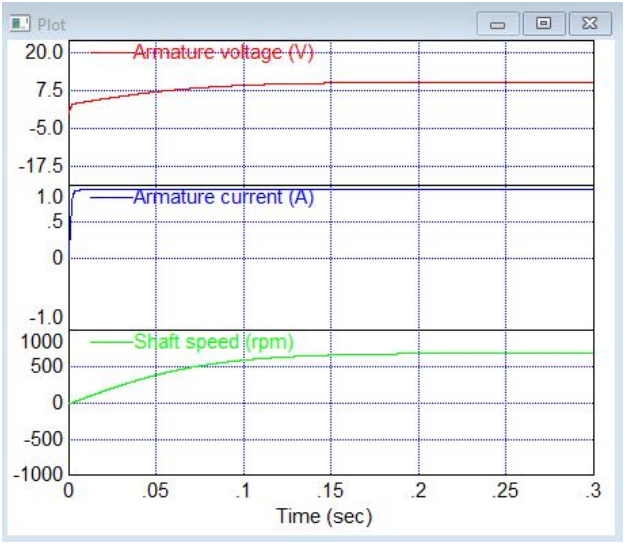


Fig. 3.56 Simulation (SIM) result, brushed DC operation, current control

The results shown in figure 3.56, confirm the presence of armature current control. Note that at start up a ramp function is used, which determines the rate of reference variable change, as set in the 'operational Variables' dialogue box. The steady state shaft speed is governed by the reference armature current setting and the applied load characteristic.

3.2.1.2 Current control-F069-boostXL8301: PIL

Reference files:

- `BrushDC-CControl-F28069M-BoostXL-DRV8301-PIL.vsm`
- `DC_Cctrl_PIL.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F28069M. This implies that the controller module shown in the simulation (see figure 3.55) is now replaced with a file: `DC_Cctrl_PIL.out` as indicated in figure 3.57.

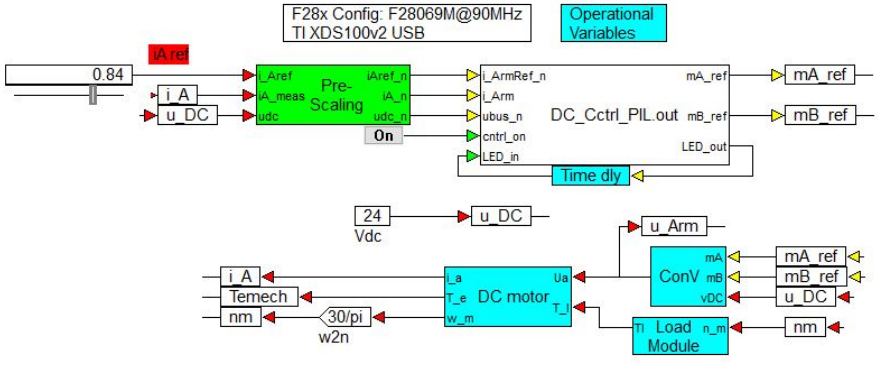


Fig. 3.57 PIL based drive, brushed DC operation, current control

The target interface model that executes the out-file, generates the modulation indices required for the converter motor and these components are not affected by the use of the PIL. Generation of the out file is based on the PIL module, see figure 3.58, which must be compiled using the command: `toolbox>codegen`.

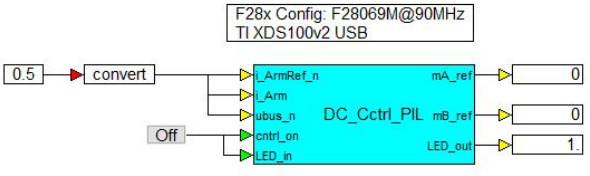


Fig. 3.58 PIL module, brushed DC operation, current control

Moving one level into the PIL module `DC_Cctrl_PIL` shows the current con-

troller as introduced in section 3.2.1.1.

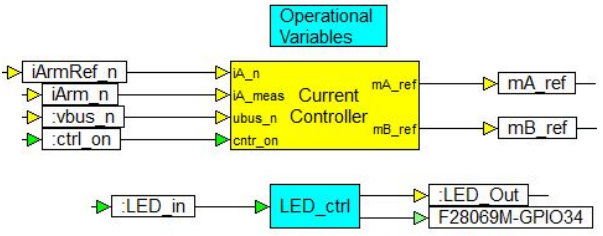


Fig. 3.59 Content PIL module, brushed DC operation, current control

Added to the current controller is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that the PIL control is operational.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational. Results produced by PIL operation are identical to the plot shown in figure 3.56 and are therefore not shown here.

3.2.1.3 Current control-F069-boostXL8301:COSIM

Reference files:

- `BrushDC-CControl-F28069M-BoostXL-DRV8301-COSIM.vsm`
- `BOOSTXL_DC_mot.psimsch`
- `DC motor model_v1.psimsch`

To evaluate operation of a controller with a switching converter, in this case the `boostXL8301`, use can be made of a `PSIMCoupler` module, as indicated in figure 3.60. This approach implies the use of PSIM together with Embed, which is referred to as 'co-simulation'.
The `main notor control` module, hatched in this case as its sampling frequency is locally set to 15 kHz, whereas the 'system' frequency (set in the 'system properties' dialogue box) is set to 20 MHz. The latter is much higher, given the need to accurately represent 15kHz voltage switching waveforms generated by the PWM unit of the converter. The content of the `main motor control` module, given in figure 3.61 shows the `current controller`, as discussed in subchapters 3.2.1.2, 3.2.1.1. However, the current and voltage measurement now makes use of the half-bridge inputs, as present in the hardware.
In this case the current shunts in the lower converter switches, as shown in figure 3.13, are used to generate the current in two half-bridges used by the brushed DC machine. After sampling and signal-processing in the converter,

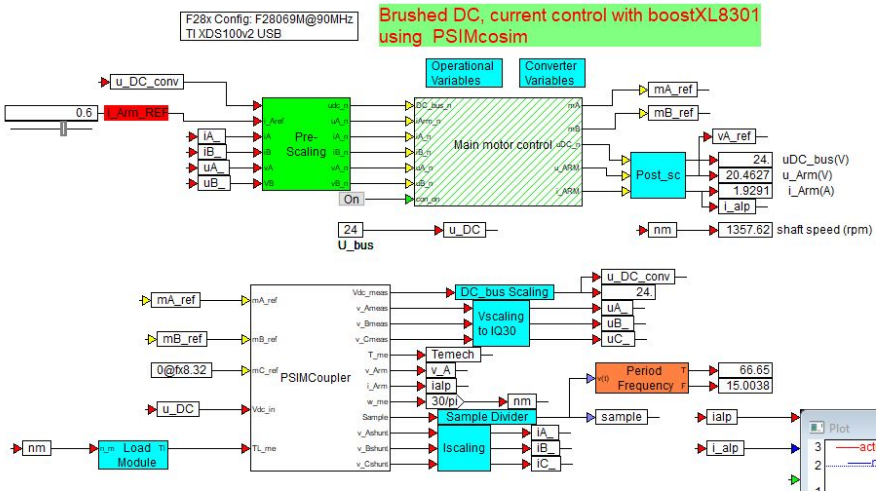


Fig. 3.60 PSIMcosim based drive, brushed DC operation, current control

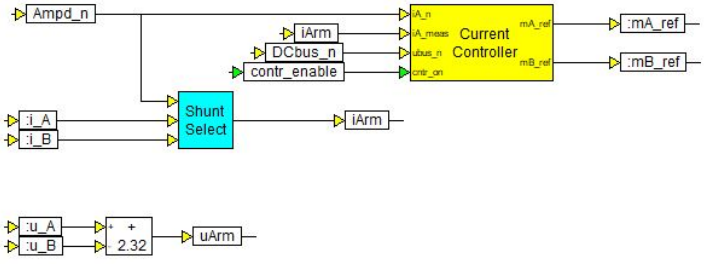


Fig. 3.61 Main motor control unit, brushed DC operation, current control

these variables appear as:`::i_A`,`:i_B` in figure 3.61. A `Shunt select` module selects the signal that represents the measured, per unit armature current `iArm`. The low pass filtered voltages (which removes the PWM component) of the two half-bridges is also present in the `main notor control` module and represented by per unit variables:`::u_A`, `:u_B` respectively. The difference between these two signals is in fact the per unit armature voltage `uArm`, which allows the user to check if this matches with the reference value.
The converter used is represented by six ideal power electronic devices as shown in figure 3.13 and each of the three converter outputs is provided with a shunt resistor that measures the phase current in a half -bridge, provided that sampling is carried at the so called 'golden moment'(see figure 2.40). Apart from the measured voltage/currents generated by the converter, the PSIM coupler unit also provides the actual motor model armature current, voltage and speed, which are presented by the variables `v_A`, `ialp` and `n_m` (after conversion to rpm) respectively (see figure 3.60).

Results achieved with the COSIM approach using the same armature voltage reference and simulation time as deployed for the SIM model (see figure 3.56) are shown in figure 3.62.

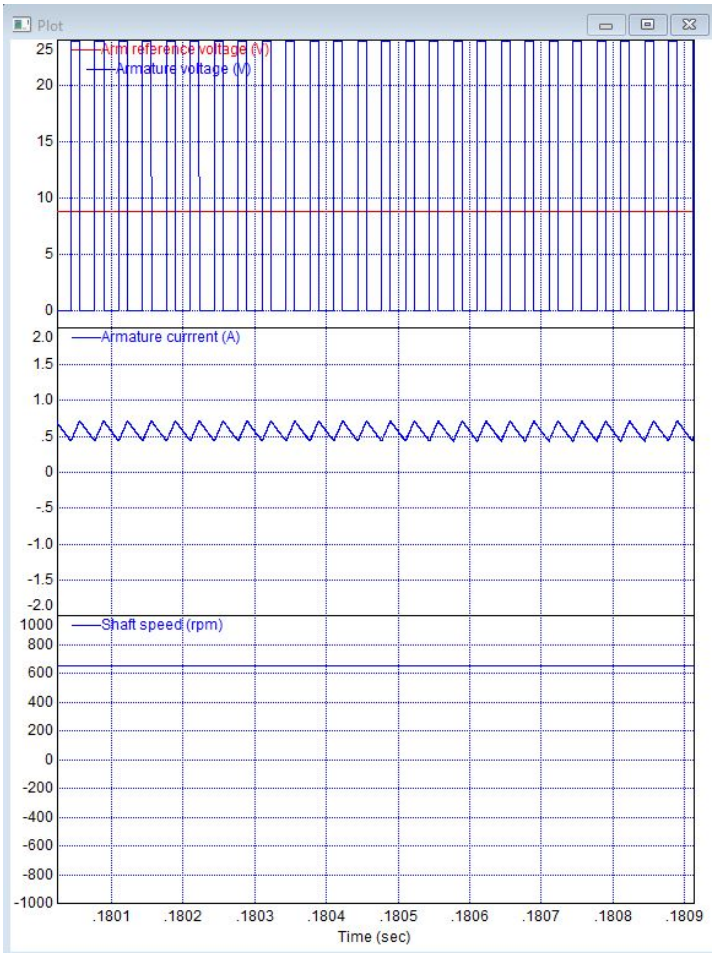


Fig. 3.62 Result COSIM, current control brushed DC

Indicated in figure 3.62 is the armature voltage waveform due to the converter together with the reference voltage for a 'zoom in' time frame around the 0.18 s time mark. This shows the switching process and the impact on the armature current, e.g presence of a ripple component due to PWM.

3.2.1.4 Current control-F069-boostXL8301: HIL

Reference files:

- `BrushDC-CurrentControl-TIF28069M_8301-compile.vsm`, for HIL-compile
- `BrushDC-CurrentControl-TIF28069M_8301-debug.vsm`,for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) as shown in figure 3.63, represents the `BOOSTXL-8301` converter (located on the `LAUNCHXL-F28069` module), which is connected to a brushed DC motor[25].

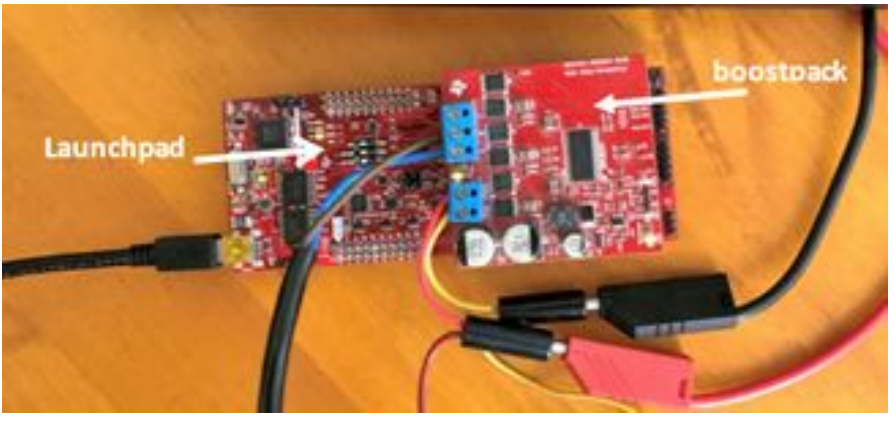


Fig. 3.63 MCU-converter setup for HIL operation

In the previous subsections, dealing with the SIM, PIL and COSIM toolchain components, only limited knowledge of the control card and converter was required. For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 3.64 shows the 'current controller' module, which must be compiled to generate an out-file .
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user reference current `iA`, which is converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Further information on this tool will be given at a later stage. Outputs of the Controller module are the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in

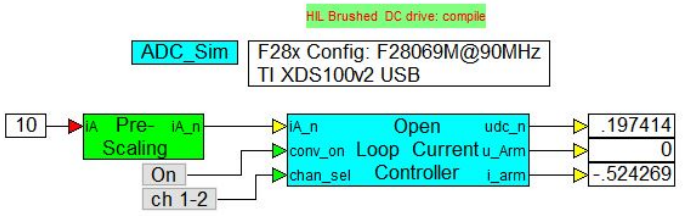


Fig. 3.64 HIL-Compile model of a brushed DC drive with current control

the HIL-debug phase) to provide a reading in Volts. In addition, the per unit armature voltage `u_Arm` and armature current `i_Arm` outputs are provided which must also be scaled in the HIL-debug stage.
Also shown in figure 3.64 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialog boxes shown in figure 3.65. In this case two dialogue boxes are used, as shown in figure 3.65, which con-

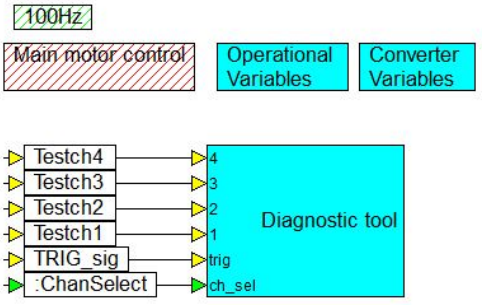


Fig. 3.65 One level down into the 'Open loop Current Controller' module

tain the parameters that need to be assigned by the user for HIL-operation. The 'operational variable' module has already been introduced in previous sub-sections and its content is shown in figure 3.66 to facilitate readability. The 'Operational Variables' dialog box, is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage. The 'Converter Variables' dialogue box, as given in figure 3.67, defines the scaling used by the Analogue to Digital converter, as discussed earlier, see

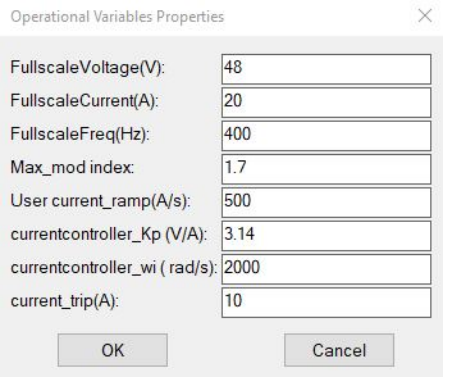


Fig. 3.66 HIL: Dialogue box 'Operational Variables'

equation (4.4). A 'Diagnostic tool' module is used to buffer two variables,

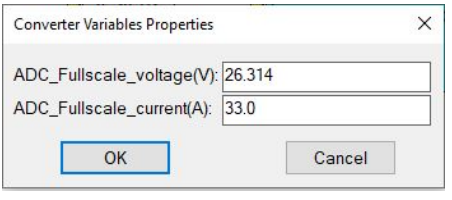


Fig. 3.67 HIL: Dialogue box 'Converter Variables'

which can subsequently displayed in a graph when operating in HIL-debug. A 'two channel' multiplexer is used to allow the user to either display variables `Test1`, `Test2` or `Test3`, `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 3.65 is the `Main motor control` module which contains the main control algorithm as needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 3.68. Of these the 'Current controller' has been discussed previously, given that it is the module used in phases : SIM, PIL and COSIM of the development toolchain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are

generated by a board specific 'ADC-PWM' module. This module is configured for with the following basic settings :

- PWM frequency: 15 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: 0.11 μs
- Board in use: `LAUNCHXL-F28069` with `BOOSTXL-8301` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed. The converter dead-time for the PWM is set to 10 MCU clock cycles. Outputs from said unit are the per unit measured DC bus voltage `uDC_n`, low-pass filtered half-bridge voltages: `uA_n`, `uB_n` and phase currents `iA_n`, `iB_n` respectively.

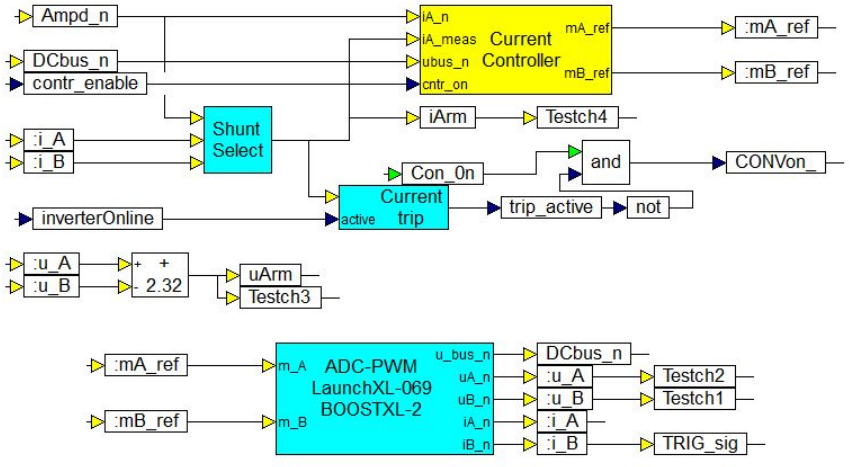


Fig. 3.68 One level down into the 'Main motor controller' module

Further information on the content of the module `ADC-PWM-LaunchXL-069 BOOSTXL-2` has been given earlier (see figure 3.23).

The HIL-debug/run tool-chain component is the operational component of the laboratory, shown in figure 3.69 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. A slider is used to set the armature current reference amplitude. Post-scaling modules are used to convert the per-unit measured DC bus voltage, armature voltage to actual voltages, as shown with numeric displays. In addition, a post-scaling unit is present to convert the measured per unit

armature current to its actual value also shown on a numeric display. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

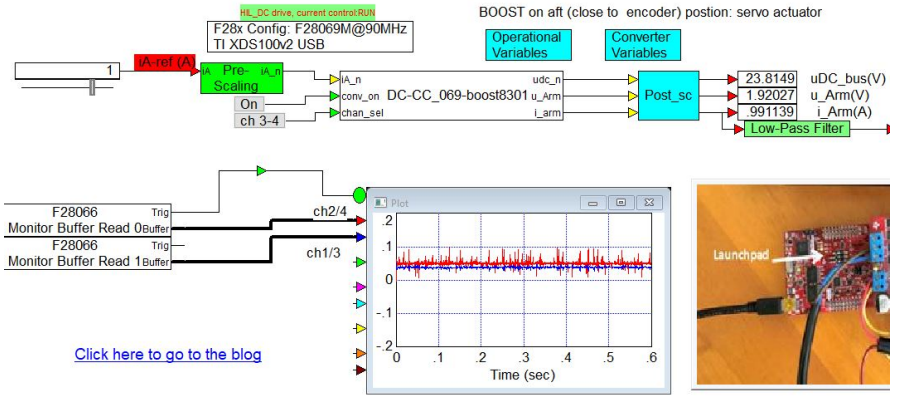


Fig. 3.69 HIL-Debug model of a brushed DC drive with current control

The plot given in figure 3.69 displays the per unit armature current (in 'red') and the per unit armature voltage, under locked rotor conditions. This requires the channel select button to be set to `ch 3-4`. Note that the current reference slider is set to `iA-ref=1.0`A, hence the per unit measured current amplitude should be $\vec{i}_n = {}^{1.0}/_{20} = 0.05$, given that a full-scale current of i_{fs}=20 V is used. Observation of the plot result confirms that the amplitude is equal to the anticipated value.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialog boxes entry specified under HIL-compile. The dialog boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).

- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive checklist, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC bus voltage level established, turn on the converter (using the `on` button, shown) and monitor the diagnostic scope, which in this example (figure 3.69) is set to show the per unit variables u^n_{arm}, i^n_{arm} on the scope module.

3.2.1.5 Current control-F0049-GANINV: HIL

Reference files:

- `C_F280049-GANINV-compile.vsm`, for HIL-compile
- `C_F280049-GANINV-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) as shown in figure 3.70 represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F280049` module) which is connected to a brushed DC motor[25]

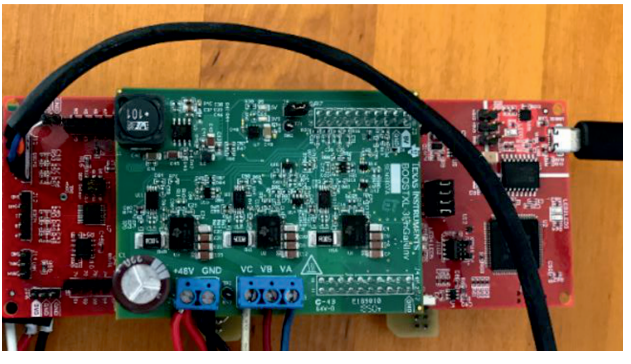


Fig. 3.70 MCU-converter setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 3.71 shows the 'current controller' module, which

must be compiled to generate an out-file.

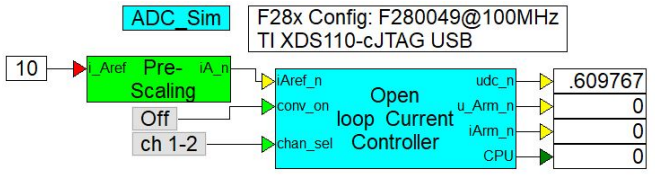


Fig. 3.71 HIL-Compile model of a brushed DC drive with current control

Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user reference current `iAref`, which is converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Further information on this tool will be given at a later stage. Outputs of the Controller module are the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts. In addition, the per unit armature voltage `u_Arm` and armature current `i_Arm` outputs are provided, which must also be scaled in the HIL-debug stage. Furthermore, a `CPU` output is added, which provides the user with a % reading of MCU activity, relative the controller sample time.
Also shown in figure 3.71 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialog boxes shown in figure 3.72.

Two dialogue boxes are used, as shown in figure 3.65, which contain the parameters that need to be assigned by the user for HIL-operation. The 'operational Variables' dialogue box, see figure 3.73, is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage.
figure 3.29.
The 'Converter Variables' dialogue box, as given in figure 3.21, defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.4). A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph when operating in HIL-debug. A 'two channel' multiplexer is used to allow the user to either display variables `Test1`, `Test2` or `Test3`, `Test4` by using a 'channel select' button discussed

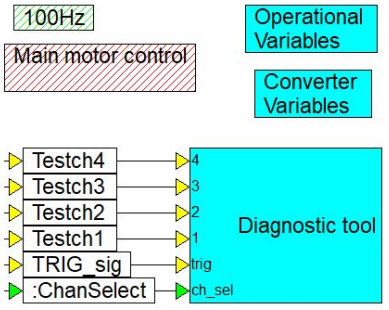


Fig. 3.72 One level down into the 'Open loop Current Controller' module

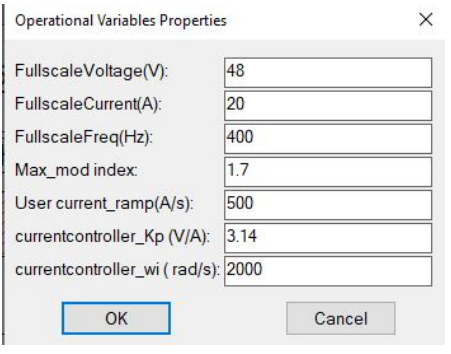


Fig. 3.73 HIL: Dialogue box 'Operational Variables'

above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 3.72 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 3.74. Of these, the 'Current controller' has been discussed previously, for a different MCU (see figure 3.55), where it was used in phases : SIM, PIL and COSIM of the respective development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are

generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F280049` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present, which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

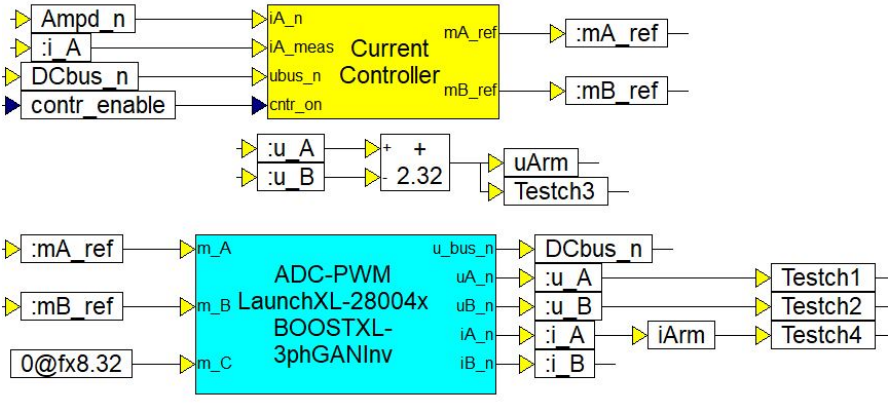


Fig. 3.74 One level down into the 'Main motor controller' module

Note that the `GANINV` converter board makes use of in-line phase current transducers, hence for this application phase 'A' unit is used directly for armature current measurement.
The `ADC-PWM LaunchXL-28004x BOOSTXL-3phGANInv` unit is used to measure the per unit, two low-pass filtered half-bridge voltages `uA_n`, `uB_n`, DC bus voltage `u_bus_n` and current `iA_n` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously.
The HIL-debug/run tool-chain component is the operational component of the laboratory, shown in figure 3.75 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes

said out-file. A slider is used to set the armature current reference amplitude. Post-scaling modules are used to convert the per-unit measured DC bus voltage,armature voltage to actual voltages, as shown with numeric displays. In addition, a post-scaling unit is present to convert the measured per unit armature current to its actual value, also shown on a numeric display. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

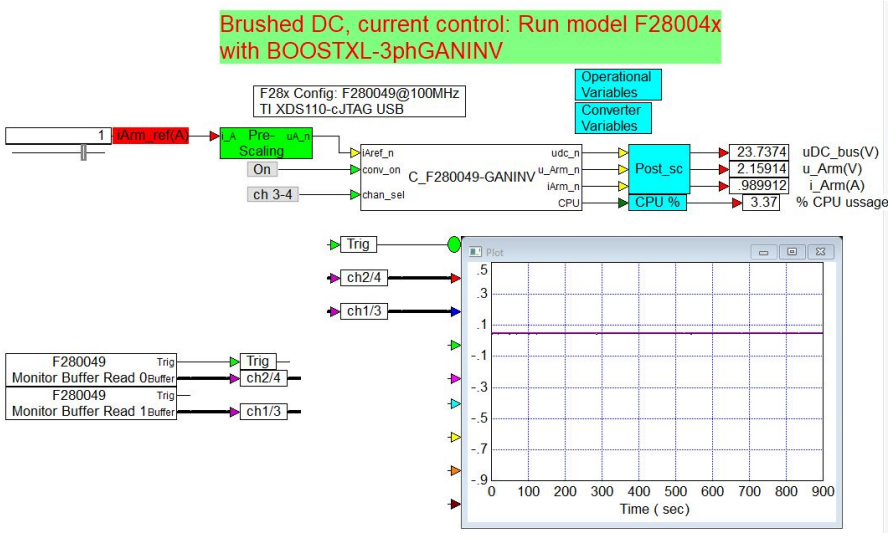


Fig. 3.75 HIL-Debug model, F0049-GANINV of a brushed DC drive with current control

The plot given in figure 3.24 displays the per unit armature current (in 'red') and the per unit armature voltage. This requires the channel select button to be set to `ch 3-4`. Note that the voltage reference slider is set to `iArm_ref=1.0`A, hence the per unit current amplitude should be $|\vec{i}|_{\text{ref}} = {}^{1.0}/_{20} = 0.05$, given that a full-scale current of i_{fs}=20 A is used. Observation of the plot result confirms that the amplitude is equal to the anticipated value. The measured armature voltage in the numerical display shows a value of 2.15(V), which corresponds to a per unit value of ${}^{2.15}/_{48} = 0.044$, given a full-scale voltage value of $v_{fs} = 48$(V).

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive checklist, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic plot, which in this example (figure 3.75) is set to show the per unit variables u^n_{arm}, i^n_{arm} on the plot module.

3.2.2 Armature Current Control-STM

Purpose of this section is to realize open loop armature current control of a brushed DC machine using ST-Micro board. The tool chain as introduced in figure 2.45 is used as a template for the ensuing subsections.

3.2.2.1 Current control-F303-ESC001: SIM

Reference file: `BrushDC-CControl-F303_ESC001-SIM.vsm`

This example considers the simulation of a brushed DC motor [25] connected to a ST-Micro `ESC001` module (see figure 3.76). The aim is to examine operation of the drive using a fixed point controller, that will subsequently be used in the HIL implementation. The motor is connected to phases 'A' and 'B', hence 'C' is not used in the example.

Drilling into the 'current controller' reveals the controller structure shown in figure 3.77. Outputs of the controller are the two modulation indices that

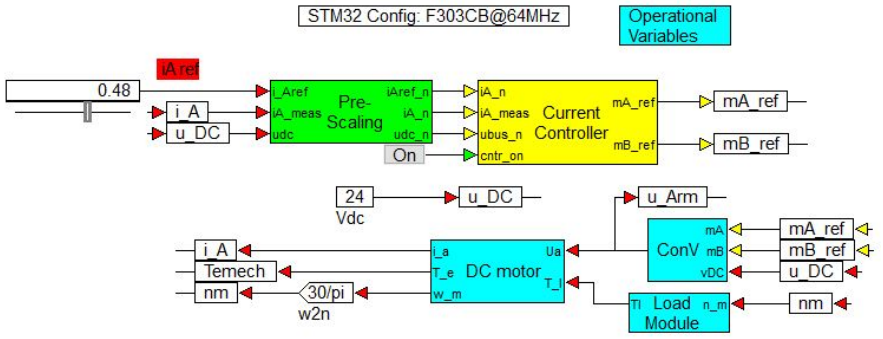


Fig. 3.76 Brushed DC drive with armature current control

controller the two converter half bridges in use. For all controllers consid-

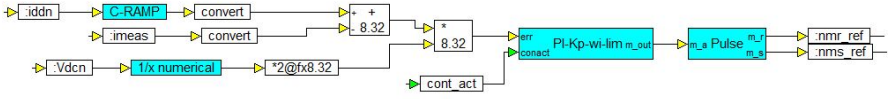


Fig. 3.77 Current controller structure

ered, the aim is to generate the modulation indices that are required by the PWM units. Consequently, it is important to convert the reference voltage to modulation index using expression (3.7).

$$m^{\mathrm{ref}} = \frac{2\,u^{\mathrm{ref}}}{u_{\mathrm{DC}}} \tag{3.7}$$

which requires access to the measured DC bus voltage u_{DC}, shown as `Vdcn` (which is the per unit value). The 'Pulse' unit splits the reference modulation index into components $\pm \frac{m^{\mathrm{ref}}}{2}$ as required to modulate the two half-bridges used. The voltage supplied to the motor `U_Arm` is found by multiplying the modulation indices `mA_ref`, `mB_ref` by half the DC bus voltage as shown in figure 3.76 and taking the difference between the two half bridge voltages (carried out by module '`ConV`'). These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables $u_a^{\mathrm{ref}}(t_k)$, $u_b^{\mathrm{ref}}(t_k)$. The aim of the switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed pre-

viously (see sub chapter 2.1.5). In this case the armature is directly connected between two converter phases and this implies that the maximum modulation index is equal to $m_{max} = 2$, which corresponds to an armature voltage of 24 V. A quadratic load is introduced, which provides a value of 100 mNm at a shaft speed of 1000 rpm. Model based current control is used, as discussed earlier in subsection 2.1.4.

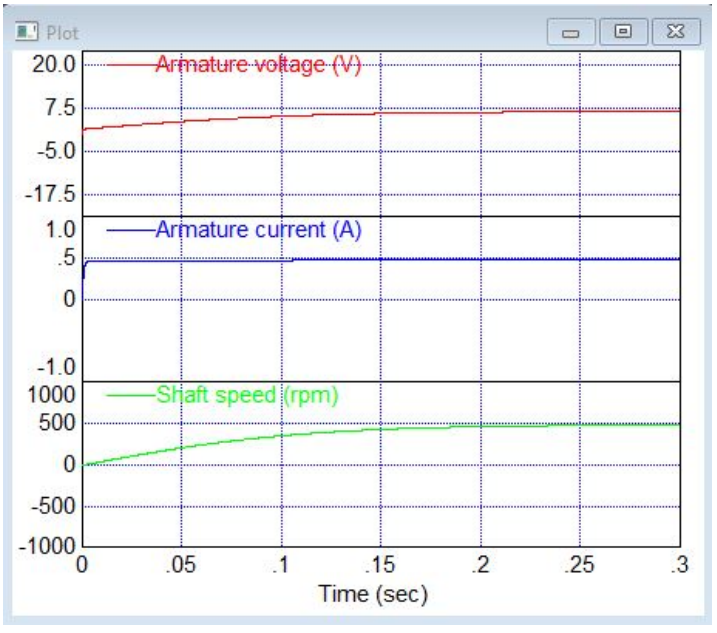


Fig. 3.78 SIM result, brushed DC operation, current control

The results shown in figure 3.78, confirm the presence of armature current control. Note that at start up a ramp function is used, which determines the rate of reference variable change, as set in the 'operational Variables' dialogue box. The steady state shaft speed is governed by the armature reference current setting and the applied load characteristic.

3.2.2.2 Current control-F303-ESC001: PIL

Reference files:

- `BrushDC-CControl-F303-ESC001-PIL.vsm`
- `DC_Cctrl_STM_PIL.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F303. This implies that the controller module shown in the simulation (see figure 3.76) is now replaced with a file: `DC_Cctrl_STM_PIL` as indicated in figure 3.79.

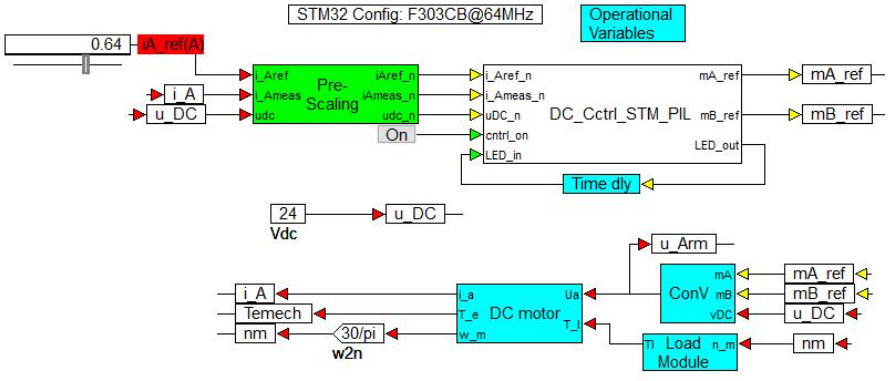


Fig. 3.79 PIL based drive, brushed DC operation, current control

The target interface model that executes the out-file, generates the modulation indices required for the converter motor and these components are not affected by the use of the PIL. Generation of the out file is based on the PIL module, see figure 3.80, which must be compiled using the command: `toolbox>codegen`.

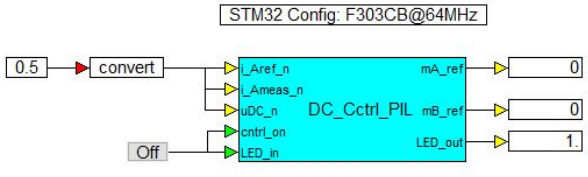


Fig. 3.80 PIL module, brushed DC operation, current control

Moving one level into the PIL module `DC_Cctrl_PIL` shows the current con-

troller as introduced in section 3.2.2.1.

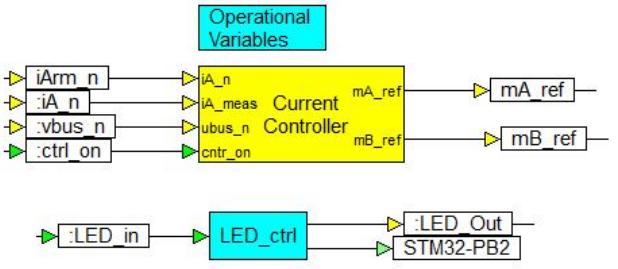


Fig. 3.81 Content PIL module, brushed DC operation, current control

Added to the current controller is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that PIL control is operational.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the ESC001 board, which indicates that the MCU is operational. Results produced by PIL operation are identical to the plot shown in figure 3.78 and are therefore not shown here.

3.2.2.3 Current control-F303-ESC001:COSIM

Reference files:

- `BrushDC-C_Control-F303-ESC001-COSIM.vsm`
- `ESC001_DC_mot.psimsch`
- `DC motor model_v1.psimsch`

To evaluate operation of a controller with a switching converter, in this case the `STM-ESC001`, use can be made of a `PSIMCoupler` module, as indicated in figure 3.82. This approach implies the use of PSIM together with Embed, which is referred to as 'co-simulation'.
The `main notor control` module, hatched in this case as its sampling frequency is locally set to 10 kHz, whereas the 'system' frequency (set in the 'system properties' dialogue box) is set to 20 MHz. The latter is much higher, given the need to accurately represent 10 kHz voltage switching waveforms generated by the PWM unit of the converter. The content of the `main motor control` module, given in figure 3.83 shows the `current controller`, as discussed in subchapters 3.2.1.2, 3.2.1.1. However, the current and voltage measurement now makes use of the half-bridge inputs, as present in the hardware.
In this case the current shunts in the lower converter switches, as shown in figure 3.43, are used to generate the current in two half-bridges used by the

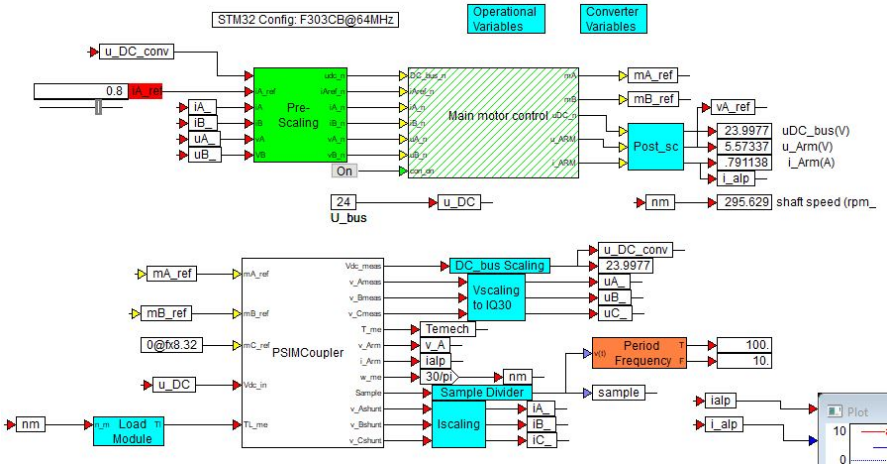


Fig. 3.82 PSIMcosim based drive, brushed DC operation, current control

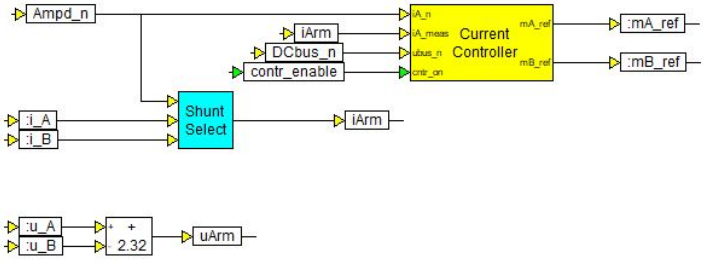


Fig. 3.83 Main motor control unit, brushed DC operation, current control

brushed DC machine. After sampling and signal-processing in the converter, these variables appear as:`:i_A`,`:i_B` in figure 3.82. A `Shunt select` module (see figure 3.83) selects the signal that represents the measured, per unit armature current `iArm`. The low pass filtered voltages (which removes the PWM component) of the two half-bridges is also present in the `main notor control` module and represented by per unit variables:`:u_A`, `:u_B` respectively. The difference between these two signals is in fact the per unit armature voltage `uArm`, which allows the user to check if this matches with the reference value.

The converter used is represented by six ideal power electronic devices as shown in figure 3.43 and each of the three converter outputs is provided with a shunt resistor that measures the phase current in a half -bridge, provided that sampling is carried at the so called 'golden moment'(see figure 2.40). Central to sensorless operation is the ability to accurately measure the phase voltages and currents. The experimental drive board(s) used in this GUIDE are mostly 'sensorless prepared', which implies that they have been purposely designed

with a data acquisition module (ADC) that undertakes the signal conditioning of the voltage/currents as needed to realize sensorless (no mechanical position encoder attached) control. In addition, access to voltages and currents, gives the user ability to estimate the motor input power, which in turn provides an indication of the DC bus current. The latter therefore, provides an estimate for the energy usage e.q. *Ah* of battery operated drives.

The 'Converter Variables' dialogue box, as given in figure 3.82, defines the scaling used by the Analog to Digital converter, which has a maximum input value of 3.3 V and is directly linked to the signal processing undertaken for voltages and currents as discussed above, as discussed earlier, see figure 3.15. Note that the ADC dialogue values, are board dependent and need to be found by inspection of the circuit (or provided). The shunt voltage(s) `v_Ashunt`; are sampled by the `Iscaling` module shown in figure 3.82. When dealing with the actual MCU, the ADC converter also takes over this sampling process. Apart from the measured voltage/currents generated by the converter, the PSIM coupler unit also provides the actual motor model armature current, voltage and speed, which are presented by the variables `v_A`, `ialp` and `n_m` (after conversion to rpm) respectively, see figure 3.82.

Results achieved with the COSIM approach using the same armature current reference as deployed for the SIM model (see figure 3.78) are shown in figure 3.84, for a zoomed in time frame around the 0.18s time mark, in order to show the converter switching process.
Indicated in figure 3.84 is the armature voltage waveform due to the converter together with the reference voltage. Co-simulation, as discussed in this sub-section, is instructive for understanding the impact of using a switching converter on drive operation.

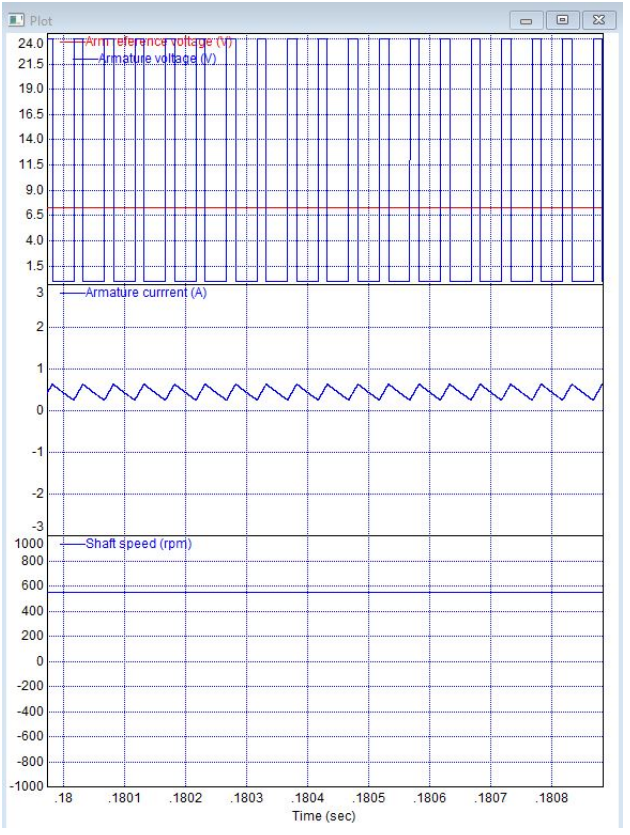


Fig. 3.84 result COSIM, current control brushed DC

3.2.2.4 Current control-F303-ESC001: HIL

Reference files:

- `DC_F303RE_ESC001_compile_HIL.vsm`, for HIL-compile
- `DC_F303RE_ESC001_run-HIL.vsm`,for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), shown in figure 3.85 represents the `STM-ESC001` module which is connected to a brushed DC motor[25]
In the previous subsections, dealing with the SIM, PIL and COSIM tool-chain components, only limited knowledge of the control card and converter was required. For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A

Fig. 3.85 MCU-converter setup for HIL operation (used here for DC motor control with phase C converter output not used)

compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 3.86 shows the 'C- controller' module, which must be compiled to generate an out-file.

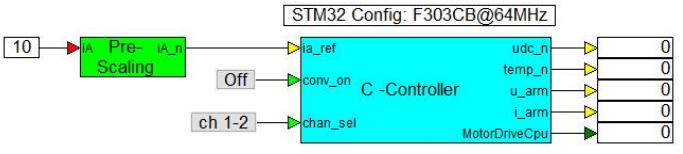


Fig. 3.86 HIL-Compile model of a brushed DC drive with current control

Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user reference current `i_A`, which is converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Further information on this tool will be given at a later stage. Outputs of the Controller module are the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts. In addition, the per unit armature voltage `u_Arm`, temperature and armature current `i_Arm` outputs are provided, which must also be scaled in the HIL-debug stage.

Moving one level down into the Controller module ('right' mouse click, after

selecting module) leads to the set of modules/dialog boxes shown in figure 3.87.

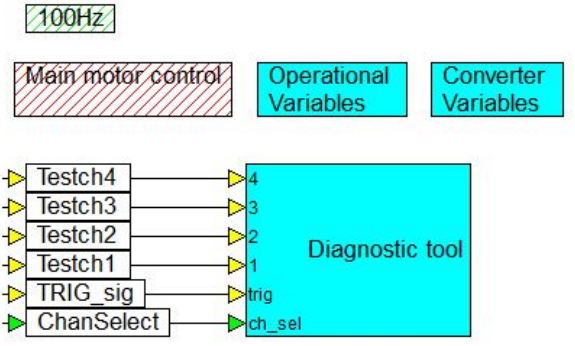


Fig. 3.87 One level down into the 'Open loop Current Controller' module

In this case two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'operational variable' module has already been introduced in previous sub-sections and its content is shown in figure 3.88 to facilitate readability.

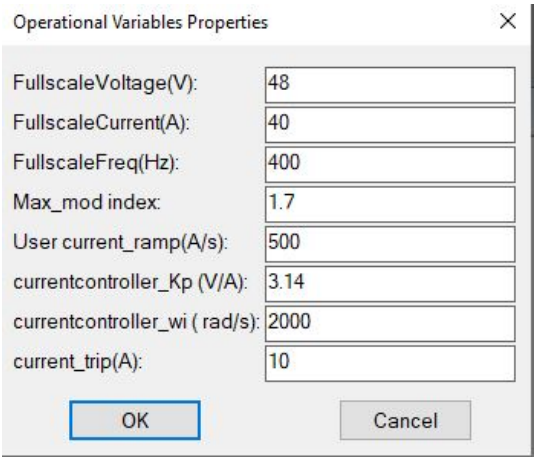


Fig. 3.88 HIL: Dialogue box 'Operational Variables'

The 'Operational Variables' dialogue box, is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage. The 'Converter Variables' box, as given in figure 3.49, defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.4).

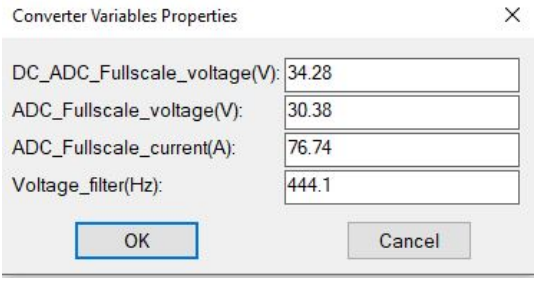


Fig. 3.89 HIL: Dialogue box 'Converter Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph when operating in HIL-debug. A 'two channel' multiplexer is used to allow the user to either display variables `Test1`, `Test2` or `Test3`, `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 3.87 is the `Main motor control` module which contains the main control algorithm as needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 3.90. Of these the 'Voltage controller' has been discussed previously, given that it is the module used in phases : SIM, PIL and COSIM of the development toolchain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured for with the following basic settings :

- PWM frequency: 10 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `STM-F303` module converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed, this implies triggering `DMA1_Channel1.0.1`.
The `ADC-PWM ESC001V1` unit is used to measure the two low-pass filtered half-

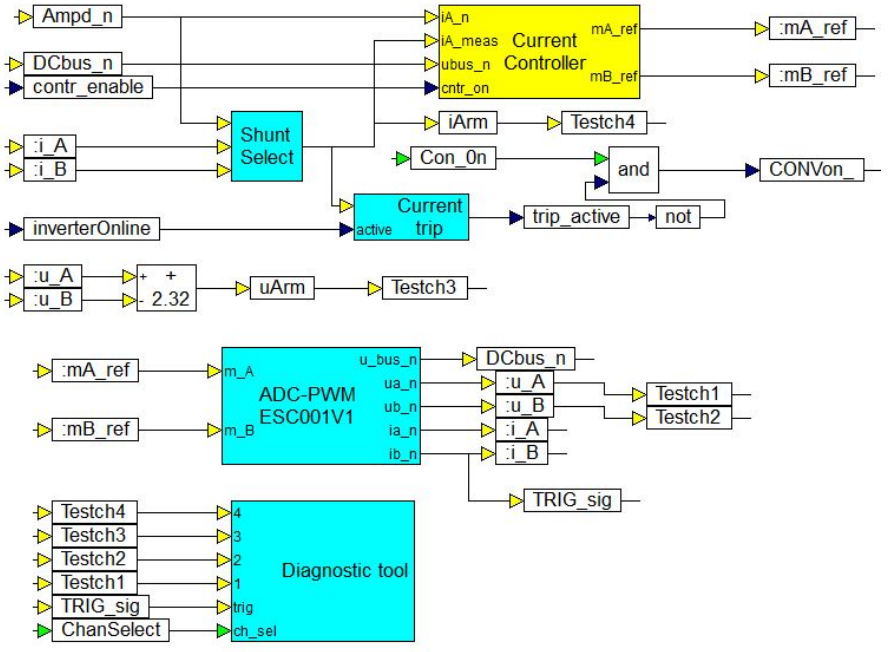


Fig. 3.90 One level down into the 'Main motor controller' module

bridge voltages `uA_N`, `uB_n` currents `iA_n`, `iB_n` and temperature respectively. ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `ubus_n`, completes the ADC measurement sequence. Also added to said module is a '`STM32-get-CPU usage` module which is used to show the % CPU activity relative the sampling time.

The HIL-debug/run tool-chain component is the operational component of the laboratory, shown in figure 3.91 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. A slider is used to set the armature voltage reference amplitude. Post-scaling modules are used to convert the per-unit measured DC bus voltage and armature voltage to actual voltages, as shown with numeric displays. In addition, the converter temperature and % CPU time, relative to the sampling time is shown. In addition, a post-scaling unit is present to convert the measured per unit armature current to its actual value also shown on a numeric display. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

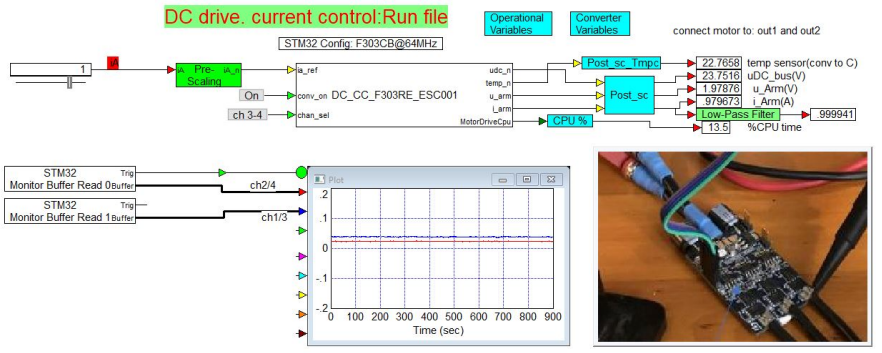


Fig. 3.91 HIL-Debug model of a brushed DC drive with current control

The plot given in figure 3.91 displays the per unit armature current (in 'red') and the per unit armature voltage. This requires the channel select button to be set to `ch 3-4`. Note that the current reference slider is set to `i_A=1.0`A, hence the per unit current amplitude should be $iA_n = {}^{1.0}/_{40} = 0.025$, given that a full-scale voltage of i_{fs}=40 A is used. Observation of the plot result confirms that the amplitude is equal to the anticipated value. Note that these results were made under locked rotor conditions.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive checklist, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not

stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic plot, which in this example (figure 3.91) is set to show the per unit variables u^n_{arm}, i^n_{arm} on the scope module.

3.3 Armature Speed Control

The theoretical basis for the speed control principles used in the SIM-HIL examples is based in the material presented in subsection 2.1.2. As such the speed control proportional and integral gains need to be defined. These are a function of the combined machine-load inertia and a user defined speed controller bandwidth, see equation (2.4).

3.3.1 Armature Speed Control-TI

Purpose of this section is to realize armature speed control of a brushed DC machine using Texas Instruments hardware. The tool chain as introduced in figure 2.45 is again used as a template for the ensuing subsections. However in this case only the tool-chain components 'simulation' (SIM) and 'Hardware in the Loop' (HIL) are discussed. For this type of control access is required to the measured shaft speed and for this purpose the encoder of the connected (to the brushed DC machine) Teknic [2] machine is used in the ensuing subsections.

3.3.1.1 Speed control-F069-boostXL8301: SIM

Reference file: `BrushDC-SpControl-F28069M-BoostXL-DRV8301-SIM.vsm`

This example considers the simulation of a brushed DC motor [25] connected to a Texas Instruments F069 Launchpad, with a three-phase boostXL8301 converter, see figure 3.92. The aim is to examine operation of the drive using a fixed point controller, that will subsequently be used in the HIL implementation. The motor is connected to phases 'A' and 'B', hence 'C' is not used in the example. As mentioned above, the encoder of a mechanically connected PM machine is used for speed measurement.

Drilling into the 'DC current controller & Spd controller' module reveals the structure shown in figure 3.93. Outputs of the controller are two modulation indices that control the two converter half bridges used. For all controllers considered, the aim is to generate the modulation indices that are required by the PWM units. Consequently, it is important to convert the reference

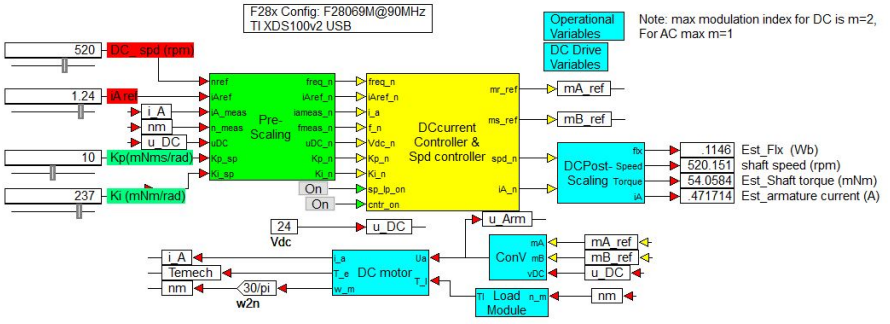


Fig. 3.92 Brushed DC drive with armature speed control

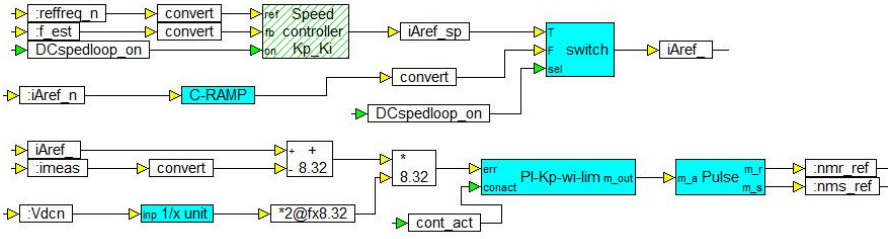


Fig. 3.93 Speed controller structure

voltage to modulation index using expression (3.8), which requires access to the measured DC bus voltage u_{DC}, shown as `Vdcn`.

$$m^{\mathrm{ref}} = \frac{2\,u^{\mathrm{ref}}}{u_{\mathrm{DC}}} \tag{3.8}$$

Model based current control, as discussed in section 2.1.4, is applied in unit `PI-Kp-wi-Lim`. Correspondingly, the gain and bandwidth of the unit are defined by the armature inductance, resistance and controller sampling frequency (see equation 2.32).
The 'Speed controller', shown 'hatched', because the sampling rate is 1 kHz, which differs from the controller sampling frequency, set to 15 kHz. Computation of the proportional K_p and integral K_i is based on the combined mechanical inertia present and the user defined speed bandwidth ω_m, as discussed earlier (see equation (2.4)).

A `switch` unit is introduced in figure 3.93, which allows the user to op-

erate with either 'current control' only, or speed control. In the latter case, the current reference for the current controller `PI-Kp-wi-lim` is provided by the speed controller, which also has a maximum (absolute) current setting defined by the user.

The 'Pulse' unit splits the reference modulation index into components $\pm \frac{m^{\text{ref}}}{2}$ as required to modulate the two half-bridges used. The voltage supplied to the motor `U_Arm` is found by multiplying the modulation indices `mA_ref`, `mB_ref` by half the DC bus voltage as shown in figure 3.92 and taking the difference between the two half bridge voltages, with the module '`ConV`'. These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables $u_a^{\text{ref}}(t_k)$, $u_b^{\text{ref}}(t_k)$. The aim of the switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sub chapter 2.1.5). In this case the armature is directly connected between two converter phases and this implies that the maximum modulation index that can be used is equal to $m_{max} = 2$, which corresponds to an armature voltage of 24 V. A quadratic load is introduced, that provides a value of 200 mNm at a shaft speed of 1000 rpm.

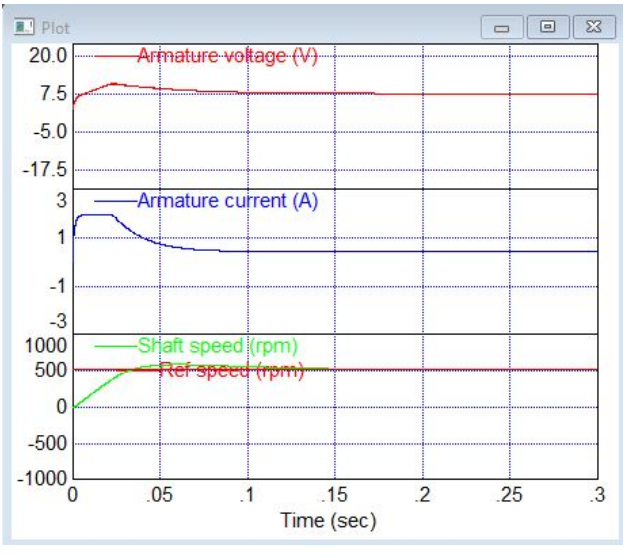


Fig. 3.94 SIM result, brushed DC operation, speed control

The results shown in figure 3.94, with speed control active, confirm the presence of armature speed control. Note that during start-up, the armature current is limited by the speed controller to 2 A, which is the 'speed-lim' value,

set in the 'DC Drive Variables' dialogue box. The steady state armature current is defined by the applied load characteristic.

3.3.1.2 Speed control-F069-boostXL8301: HIL

Reference files:

- `BrushDC-Sp&CurrentControl-F28069M_8301-compile.vsm`, for HIL-compile
- `BrushDC-Sp&CurrentControl-F28069M_8301-debug.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), as shown in figure 3.95, represents the `BOOSTXL-8301` converter (located on the `LAUNCHXL-F28069` module) which is connected to a brushed DC motor[25].

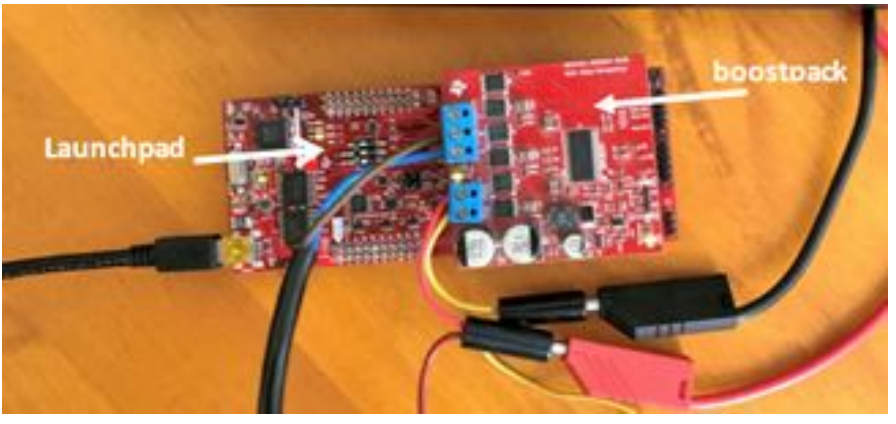


Fig. 3.95 MCU-converter setup for HIL operation

Mechanically attached to the brushed DC is a PM Teknic machine, of which the 'J4' encoder cable is connected to 'QEP2' on the launchpad. This means that the encoder of said machine is used to generate a measured shaft speed signals for the speed controller. In the previous subsections, dealing with the SIM, PIL and COSIM toolchain components, only limited knowledge of the control card and converter was required.
For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches, must be considered carefully.
A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given

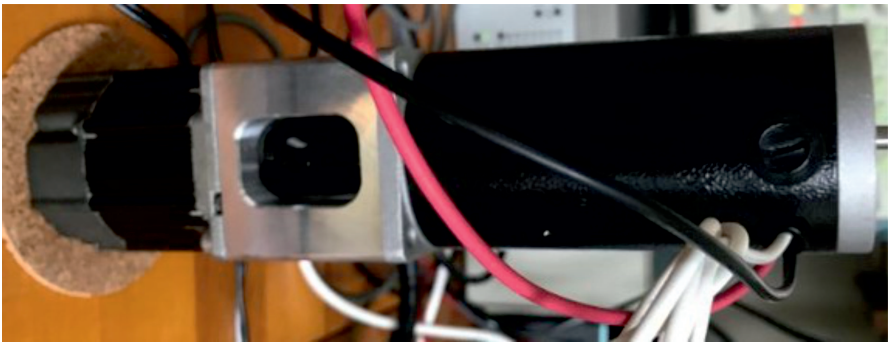


Fig. 3.96 DC motor connected to PM for HIL operation

controller configuration. The model given in figure 3.97 shows the 'Speed & current controller' module, which must be compiled to generate an out-file.

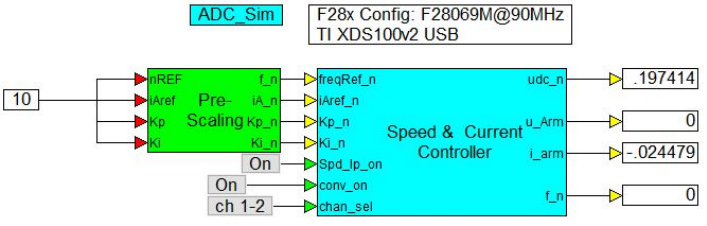


Fig. 3.97 HIL-Compile model of a brushed DC drive with speed &current control

Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user reference current `iAref`, shaft speed reference (rpm) `nREF` and proportional/integral speed gains of the speed controller, which need to be converted to 'per unit' values in the 'pre-scaling' module. Three logic inputs are provided, of which two are used to turn on the converter and to select which combination of diagnostic channels should be viewed. A third logic input `Sp_lp_On` is used to select speed or current control. Further information on this tool will be given at a later stage.
Outputs of the Controller module are the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts. In addition, the per unit armature voltage `u_Arm`,armature current `i_Arm` and shaft speed `f_n` outputs are provided, which must also be scaled in the HIL-debug stage.

Also shown in figure 3.97 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach

is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 3.98.

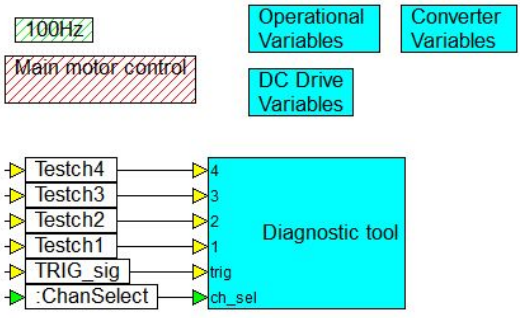


Fig. 3.98 One level down into the 'Speed & Current Controller' module

In this case two dialogue boxes are used, as shown in figure 3.98, which contain the parameters that need to be assigned by the user for HIL-operation. The 'operational variable' module has already been introduced in previous sub-sections and its content is shown in figure 3.99 to facilitate readability.

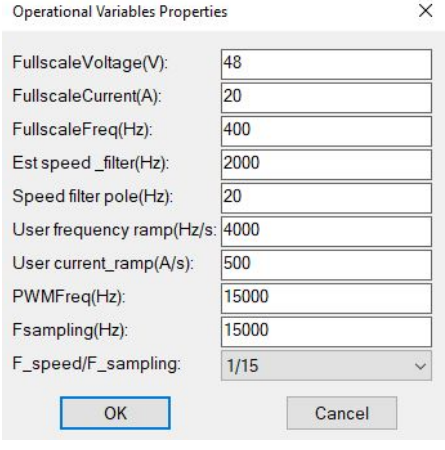


Fig. 3.99 HIL: Dialogue box 'Operational Variables'

The 'Operational Variables' dialogue box, is used to assign full scale values for

the voltage, current, frequency and the ramp change for the reference frequency and current. Furthermore, the `Est-speed_filter` and `speedfilter pole` values are introduced in this dialog box, which are used for shaft speed estimation.

Also shown in figure 3.98 is a dialogue module 'DC Drive Variables'(see figure 3.100), which is used to set the current controller gain/bandwidth together with the chosen maximum modulation index (therefore, the upper absolute voltage limit, set to $1.8 * 12 = 21.6$V). This module, also sets the absolute maximum armature current for the speed controller and the flux level on the DC machine (given the need to provide an estimate for the shaft torque).

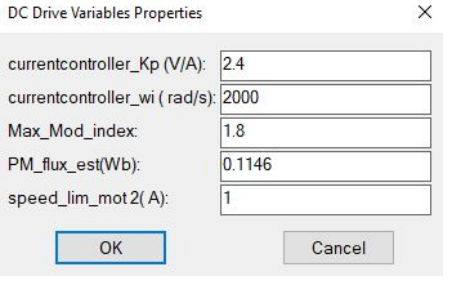


Fig. 3.100 HIL: Dialogue box 'DC Drive Variables'

The 'Converter Variables' dialogue box, as given in figure 3.101, defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.4).

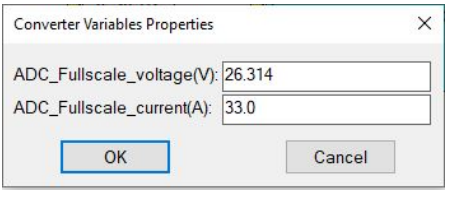


Fig. 3.101 HIL: Dialogue box 'Converter Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph when operating in HIL-debug. A 'two channel' multiplexer is used to allow the user to either display variables `Test1`, `Test2` or `Test3`, `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 3.98 is the `Main motor control` module which contains the main control algorithm as needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 3.102. Of these the 'DC current & spd controller' has been discussed previously, given that it is the module used in phase : SIM of the development toolchain.

The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured for with the following basic settings :

- PWM frequency: 15 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: 0.11 μs
- Board in use: `LAUNCHXL-F28069` with `BOOSTXL-8301` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed. The converter dead-time for the PWM is set to 10 MCU clock cycles. Outputs from said unit are the per unit measured DC bus voltage `uDC_n`, low-pass filtered half-bridge voltages: `uA_n`, `uB_n` and phase currents `iA_n`, `iB_n` respectively.
Further information on the content of the
module `ADC-PWM-LaunchXL-069 BOOSTXL-2` has been given earlier (see figure 3.23). Also shown in figure 3.102 is the `Encoder speed/angle` module, which processes the encoder signals from the PM machine that is attached to the brushed DC machine. A module `Speed_estimator` is used to derive a per unit shaft speed estimate `fe_n` as required for the speed controller.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 3.103 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. A slider is used to set the armature current reference amplitude. Post-scaling modules are used to convert the per-unit measured DC bus voltage, armature voltage to actual voltages, as shown with numeric displays. In addition, a post-scaling unit is present to convert the measured per unit armature current to its actual value also shown on a numeric display. Two

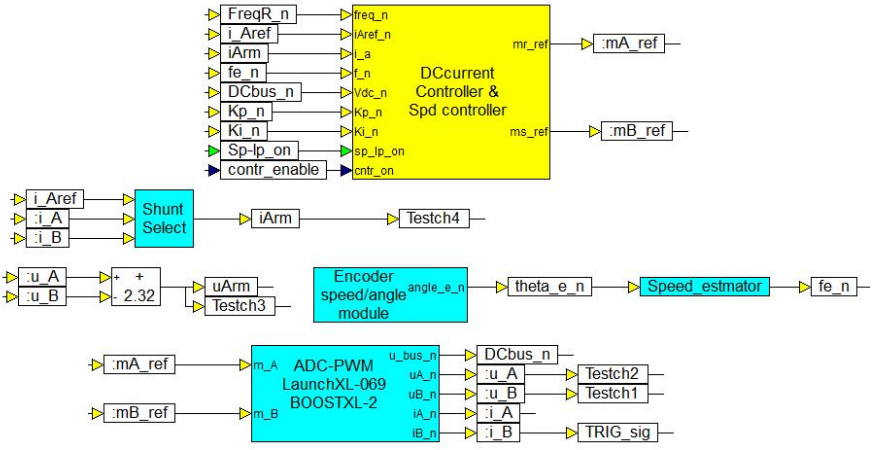


Fig. 3.102 One level down into the 'Main motor controller' module

'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

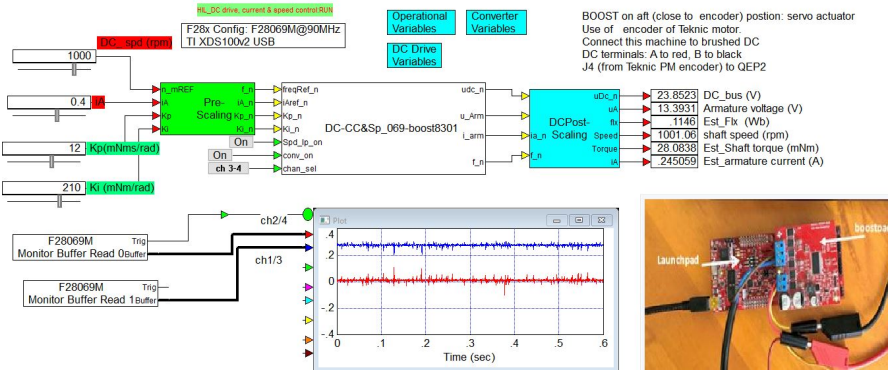


Fig. 3.103 HIL-Debug model of a brushed DC drive with speed &current control

The plot given in figure 3.103 displays the per unit armature current (in 'red') and the per unit armature voltage. This requires the channel select button to be set to `ch 3-4`. Note that speed control is active and the speed reference slider is set to `n-ref=1000`rpm, hence the measured shaft speed (on the numeric display) amplitude should match with said reference value. Speed control under no-load is shown, hence the armature current is almost zero. The armature voltage is speed dependent and the per unit value is shown on the plot.

Prior to activating this type of laboratory, the reader needs to be aware

of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not, stop the program and restart. With the correct DC bus voltage level established, turn on the converter (using the `on` button, shown) and monitor the diagnostic scope, which in this example (figure 3.103) is set to show the per unit variables u^n_{arm}, i^n_{arm} on the plot module.

Chapter 4
PMsyn and BLDC drives

This chapter is concerned with the control of 'Permanent Magnet Synchronous Machines' PMsyn (also referred to as PMSM) and 'Brushless DC' BLDC drives. The developmental tool-chain shown in figures 2.43, 2.45 is followed in this chapter. This implies that in most cases the toolchain: 'Simulation' (SIM), 'Processor in the Loop'(PIL),'CO-simulation' (COSIM) and 'Hardware in the Loop'(HIL) is discussed for Texas Instruments (TI) and ST-Micro (STM) based hardware examples. The drive control options for PM drives are: 'Voltage & Frequency Control','Current & Frequency Control' and'Field Oriented Control' (FOC) sensored and sensorless (no encoder). Furthermore, Field weakening is also discussed for FOC sensored/sensorless examples. With respect to sensorless FOC, both FAST[23] and 'Altair Flux tracker' (AFT) examples are shown. In addition, multiple HIL examples are introduced with different hardware, to reduce drive development time.

4.1 Voltage & Frequency Control

In this case the objective is to create a controller that can generate a rotating voltage space vector as shown in figure 4.1. For simplicity the reference vector $\vec{u}^{\text{ref}}$ is tied to the 'd-axis' of a rotating 'd,q' coordinate system (hence u_q is set to zero for convenience) of which the user can set the rotational speed defined by the reference frequency f^{ref}. Also shown in figure 4.1 is an 'α, β' orthogonal coordinate system that is tied to the machine stator, hence stationary. The amplitude of the reference vector u^{ref} can also be set by the user, where care should be taken, as the currents encountered will be voltage/frequency dependent. The angle between the 'd-axis' and stationary 'α-axis', is defined as θ, which in turn is the integral of said reference frequency (in rad/s). Note that angle θ does NOT represent the orientation of the flux vector, as these examples are concerned with open loop voltage & frequency control. Consequently, the PM flux vector will be aligned with the current vector, when

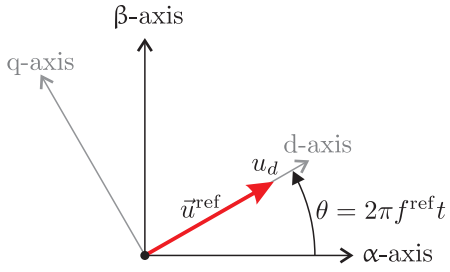


Fig. 4.1 Vector diagram: open loop voltage & frequency control

operating under no-load (as is the case here). Note also that these examples are based on the use of either Texas Instruments or ST Micro hardware.

4.1.1 Voltage & Frequency Control-TI

Purpose of this section is to realize voltage & frequency control of a PMsyn machine using Texas Instruments hardware. At the same time this type of example is used to make a functional check on the current/voltage measurements and overall functioning of the drive board and software. As such this section provides a set of 'foundation' examples, which will substantially reduce the development time required for a new motor drive. The reason being that users can simply redesign/modify the controller to their needs as the hardware setup phase for HIL's provided is in place. The tool chain as introduced in figure 2.45 is used as a template for the ensuing subsections.

4.1.1.1 Voltage & Frequency control-F0049-GANINV: SIM

Reference file: `V&F_0049_GANINV-SIM.vsm`

This example considers the simulation of a three-phase permanent magnet PMsyn Teknic motor [2] connected to a Texas Instruments F0049 GANINV converter, as shown in figure 4.2. The aim is to examine operation of the drive using a fixed point controller, that will subsequently be used in the HIL implementation.

Drilling into the 'voltage controller' reveals the controller structure shown in figure 4.3. Outputs of the controller are the three modulation indices that control the three converter half bridges. For all controllers considered the aim is to generate the modulation indices that are required by the PWM units.

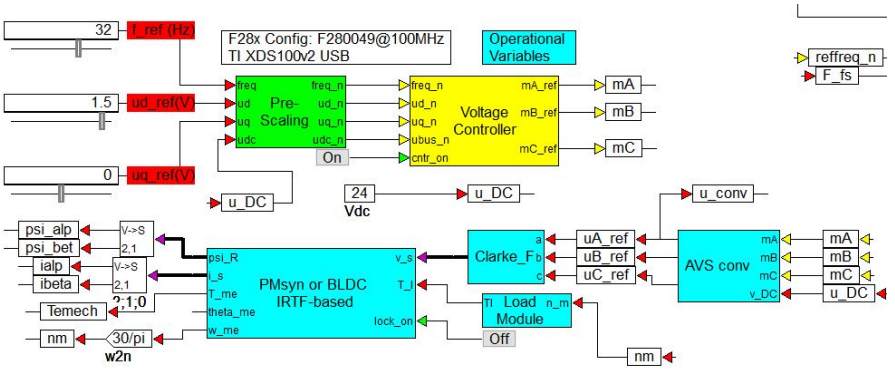


Fig. 4.2 Voltage & Frequency control: simulation

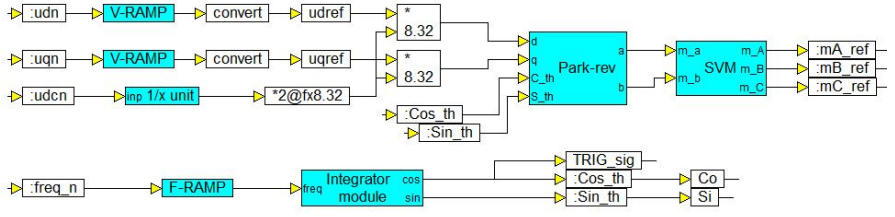


Fig. 4.3 Voltage controller structure

Consequently, it is important to convert the reference voltage to modulation index using expression (4.1).

$$m^{\text{ref}} = \frac{2\,u^{\text{ref}}}{u_{\text{DC}}} \tag{4.1}$$

This expression requires access to the measured DC bus voltage u_{DC}, shown as `udcn` in figure 4.3. The space vector modulation 'SVM' unit centers the active vectors, relative to the sampling interval, as discussed in section 2.1.5. The voltages supplied to the motor `u_A_ref`,`u_B_ref`,`u_C_ref` are found by multiplying the modulation indices `mA`, `mB`, `mC` by half the DC bus voltage as executed in module `AVS conv`, see figure 4.2. These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables. The aim of the switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sub chapter 2.1.5). A quadratic load is introduced, which provides a value of 10 mNm at a shaft speed of 1000 rpm.

The IRTF based PM model [13] has a user option to select a synchronous or BLDC type machine, as discussed earlier (see figure 2.12). When selecting the 'BLDC' option a six harmonic component is added to the PM flux locus.

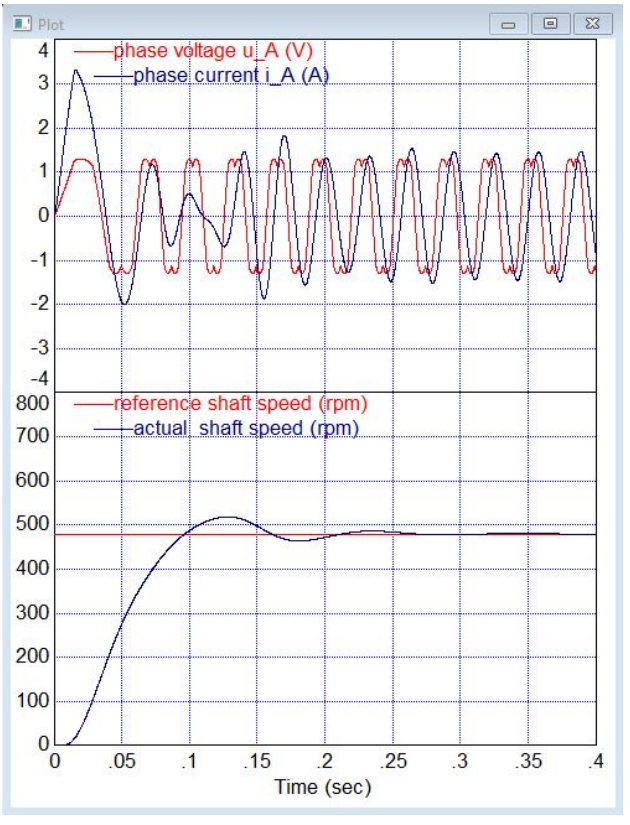


Fig. 4.4 SIM result, PM syn operation, voltage & frequency control

The results shown in figure 4.4, confirm the presence of voltage and frequency control. Note that at start up, ramp functions are used, which determine the rate of reference voltage and frequency variable changes (set by the user in the 'Operational Variables' dialogue box).
Of importance is to consider the vector plot (see figure 4.5), which shows the orientation of the unity d-axis ('green'), voltage vector ('blue') and current vector ('red') at time mark ≈ 0.4s. Observation of the voltage vector confirms that it is indeed aligned with the d-axis and its amplitude is 1.5 V, as expected because `uq_ref` is set to zero. It is also of interest to observe that the phase

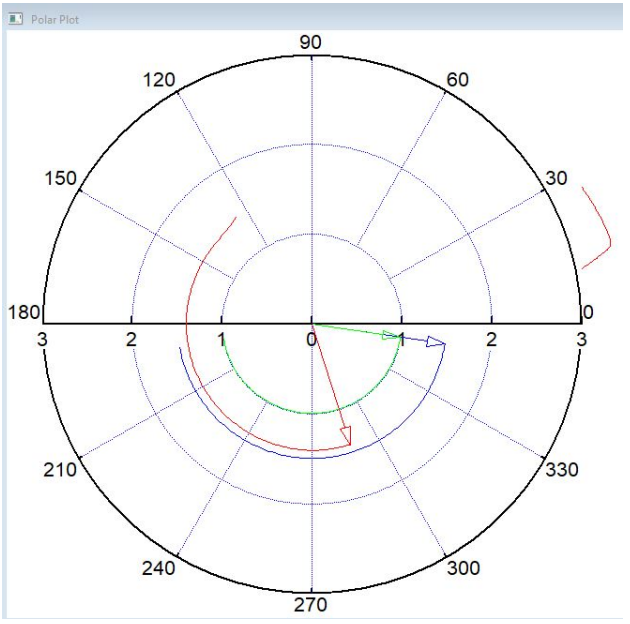


Fig. 4.5 SIM result, PM syn operation, voltage & frequency control:vector plot

voltage amplitude is a factor ($\sqrt{2}/3$) lower than the reference voltage amplitude, which is advantageous, as discussed in sub chapter 2.1.5. Furthermore, the current vector lags the voltage vector, which is expected due to the inductive nature of the machine. Note also that the PM flux vector will be aligned with the current vector, when operating under no-load conditions.

4.1.1.2 Voltage & Frequency control-F0049-GANINV: PIL

Reference files:

- `V&F_F0049-GANINV-PIL.vsm`
- `V&F_PIL_mod.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F280049. This implies that the controller module shown in the simulation (see figure 4.2) is now replaced with a file: `V&F_PIL_mod.out`, as indicated in figure 4.7.
The target interface model that executes the out-file, generates the modulation indices required for the converter motor and these components are not

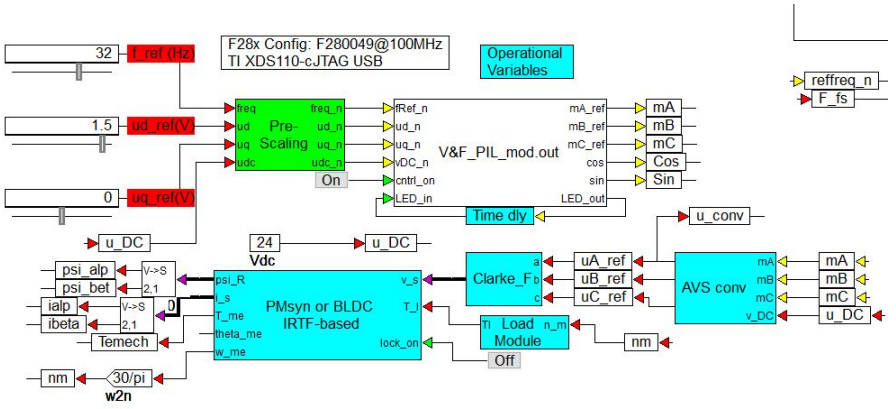


Fig. 4.6 PIL based drive, PM syn operation, voltage & frequency control

affected by the use of the PIL. Generation of the out-file is based on the PIL module (see figure 4.7), which must be compiled using the command: `toolbox>codegen`.

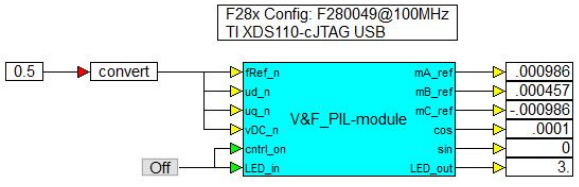


Fig. 4.7 PIL module, voltage & frequency control

Moving one level into the PIL module `V&F_PIL_mod` (see figure 4.8) shows the voltage controller as introduced in section 4.1.1.1.

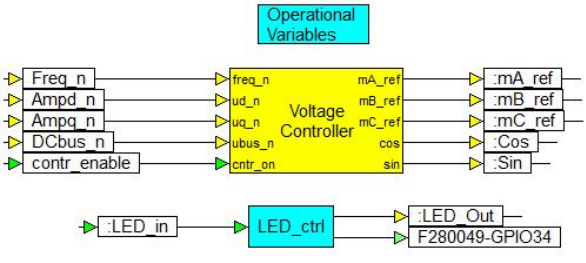


Fig. 4.8 Content of PIL module, voltage & frequency control

Added to the PIL voltage controller is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that PIL control is operational. In addition, two variables `Cos`, `Sim` are provided as outputs and these are used in a vector plot to show the 'd' axis of the controller algorithm.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational. Results produced by PIL operation are identical to the plots shown in figures 4.4, 4.5 and are therefore not shown here.

4.1.1.3 Voltage & Frequency control-F0049-GANINV:COSIM

Reference files:

- `V&F_0049_GANINV-COSIM.vsm`
- `GAN_INVv12.psimsch`
- `my motor model_v3.psimsch`

To evaluate operation of a controller with a switching converter, in this case the `GANINV`, use can be made of a `PSIMCoupler` module, as indicated in figure 4.9. This approach implies the use of PSIM together with Embed, which is referred to as 'co-simulation' (COSIM).

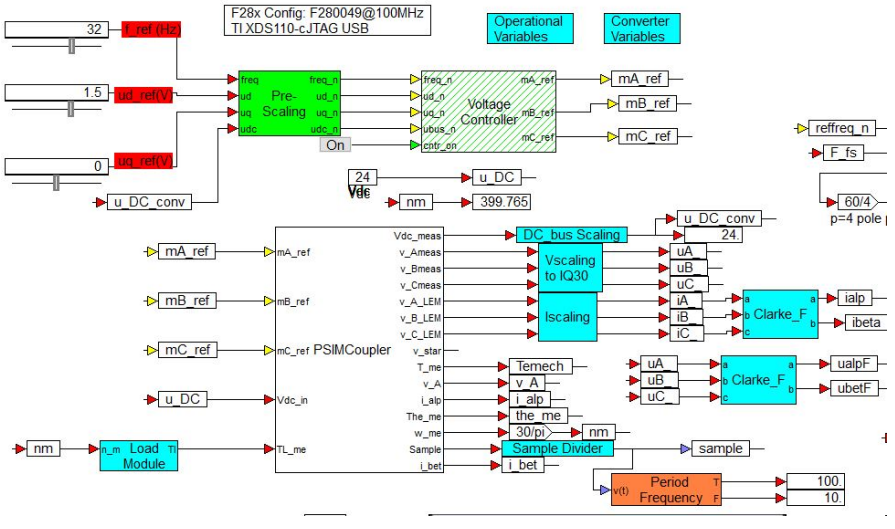


Fig. 4.9 COSIM based drive, voltage & frequency control

The `Voltage Controller` module is hatched in this case, as its sampling frequency is locally set to 10 kHz, whereas the 'system' frequency (set in the 'system properties' dialogue box) is set to 20 MHz. The latter is much higher, given the need to accurately represent 20 kHz voltage switching waveforms generated by the PWM unit of the converter. The content of the `Voltage controller` module was introduced earlier, see figure 4.3. However, the current and voltage measurements now make use of the signal processing circuitry connected to the half-bridges.

In this case the current transducers are in the converter phases, as shown in figure 4.10. After sampling and signal-processing in the converter, these variables appear as: `iA_`, `iB_`, `iC_` in figure 4.9. The low pass filtered voltages (which removes the PWM component) of the three half-bridges is also present in figure 4.9 and are also scaled and sampled by the module `Vscaling to IQ30`. In this case the per unit variables are re-scaled to actual values: `uA_`, `uB_` and `uC_` respectively.

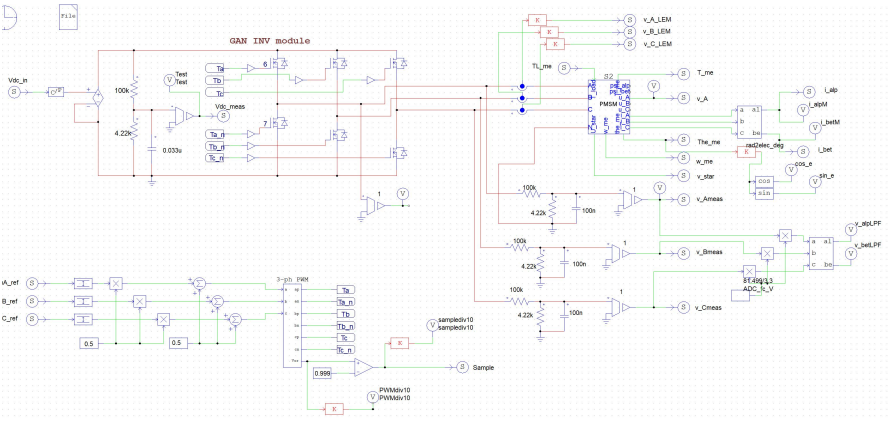


Fig. 4.10 Inside the PSIM coupler module

Further observation of figure 4.10 shows the presence of a module `PMSM`, which is a generic IRTF based model of an IPM synchronous machine machine, as shown earlier (see figure 2.17). However, the PSIM model, shown in figure 4.11, has an 'electrical interface' to allow use of said model with a converter. Note that in this case an IPM is used (because this model has the option of setting unequal L_d, Lq values if required) and this is simply a non-salient PM (PMsyn) in this example, because the inductance variables are set to $L_d = L_s$, $L_q = L_s$ respectively. Under those conditions, the IPM model is identical to the PMsyn model shown in figure 2.14.

A PWM unit referred to as, `3-ph PWM` (see figure 4.10), generates the gate control signals used to switch on/off the upper and lower devices of each

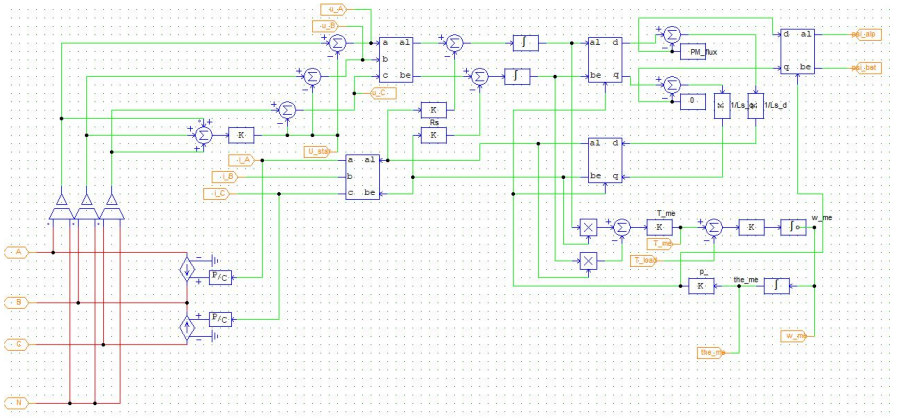


Fig. 4.11 PSIM sub-module: 'PMSM' which represents an IRTF based IPM machine model

converter leg. Connected to the output of each converter terminal is a resistance/capacitor network that is used to measure and low-pass filter the converter output voltage in order to eliminate the PWM switching effects on the motor voltage measurement signals, as mentioned earlier.

Central to sensorless operation is the ability to accurately measure the phase voltages and currents. The experimental drive board(s) used in this GUIDE are mostly 'sensorless prepared', which implies that they have been purposely designed with a data acquisition module (ADC) that undertakes the signal conditioning of the voltage/currents as needed to realize sensorless (no mechanical encoder attached) control. In addition, access to voltages and currents, gives the user ability to estimate the motor input power, which in turn provides an indication of the DC bus current. The latter therefore, provides an estimate for the energy usage e.q. Ah of battery operated drives.

The converter model example discussed here is 'sensorless prepared' and at this stage it is helpful to examine the data signal condition of the circuit in more detail, where use is made of figure 4.12. This is required because a number of critical hardware based parameters must be provided to the controller, by the user, to ensure correct operation of the drive.
Readily apparent in figure 4.12 is the resistor/capacitor network used to attenuate and low-pass filter the converter output voltage to be used by the controller. The attenuation/gain G_R and corner frequency ω_c of the low pass filter circuit are defined by expression (4.2).

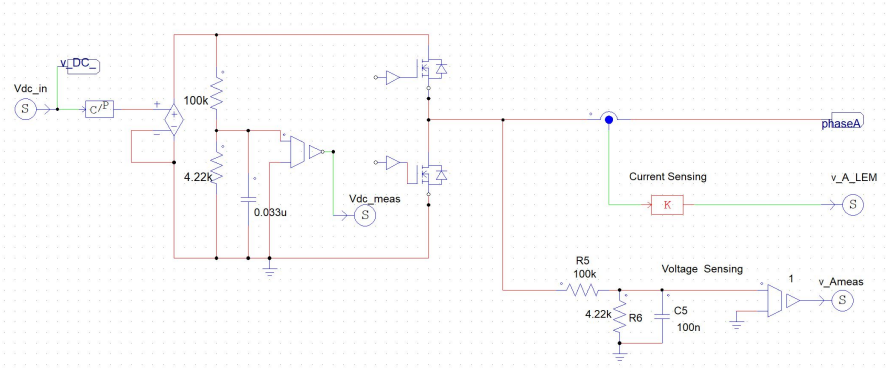


Fig. 4.12 One (of three) phase PSIM model of the GANINV unit

$$G_R = \frac{R_6}{R_5 + R_6} \tag{4.2a}$$

$$\omega_c = \frac{R_5 + R_6}{R_5\, R_6\, C_5} \tag{4.2b}$$

The corner frequency ω_c (rad/s) and attenuation of this filter are board dependent, hence their value must be identified by inspection of the components (or provided via the circuit diagram). The example given in figure 4.12 shows the voltage filter components as present in the `GANINV` module, which provide an attenuation of $G_R = 0.0409$ and $\omega_c = 2469$ rad/s respectively. Note that the three capacitors used in the low-pass filters have been changed on the GANINV board from 33nF to 100nF, which is more effective for sensorless applications. The low-pass filtered converter voltage(s) `v_Ameas` are sampled by the `Vscaling to IQ30` module shown in figure 4.9. When dealing with the actual MCU, the ADC converter takes over this sampling process. Measurement of the converter DC bus voltage is similar to that undertaken for the converter voltages.

An important part of the data acquisition process is measurement of the phase current using an in-line shunt resistor `R_mathrm{RC}` in the converter and represented by a 'transducer' (see figure 4.12). Furthermore, the current shunt voltage is amplified by an op-amp with gain G_{sh}, hence the voltage output `v_A_LEM` for the MCU (and an output of the PSIM module) is equal to

$$u_{\mathrm{LEM}} = i_{\mathrm{conv}}\, R_{\mathrm{RC}}\, G_{\mathrm{sh}} \tag{4.3}$$

The 'Converter Variables' dialogue box (see figure 4.9), defines the scaling used by the Analog to Digital converter, which has a maximum input value of 3.3 V and is directly linked to the signal processing undertaken for voltages and currents, as discussed above. For example, the 'ADC-fullscale-voltage' value is the theoretical peak-to peak input voltage which must be applied to the low-pass filter network (see figure 4.12) to ensure that the ADC input level varies between $0 \rightarrow 3.3$V. Subsequent use of equations (4.2),(3.3) shows that these values are defined by expression (4.4).

$$\mathrm{ADC-fullscalevoltage} = \frac{3.3}{G_{\mathrm{R}}} \tag{4.4a}$$

$$\mathrm{ADC-fullscalecurrent} = \frac{3.3}{R_{\mathrm{RC}}\, G_{\mathrm{sh}}} \tag{4.4b}$$

where G_{R}, is the low-pass filter attenuation used for measuring the converter voltages. Similarly, the 'ADC-fullscale-current' value is the peak to peak current in the shunt resistance, which corresponds to an ADC input voltage range of $0 \rightarrow 3.3$ V.
Hence its value is determined by the shunt resistance value R_{RC} and the signal amplification gain G_{sh}, between said resistor and ADC input. The voltage and current measurements are unipolar and bipolar respectively. This implies that the voltage measurement is referenced to the negative converter DC bus voltage, whilst the current is referenced to the center of the ADC voltage span, i.e $^{3.3}/_{2}$ V, as provided by an op-amp. Note that the ADC fullscale current/voltage dialogue values, are board dependent and need to be found by inspection of the circuit (or provided).

The importance of using appropriate values for the ADC full-scale values should not be underestimated, as this determines the utilization of the number of bits used in a drive application. For example, a 12 bit ADC converter, as often used, has a $2^{12} - 1 = 4095$ resolution [14]. This means that the voltage/current per bit for a 12bit ADC unit can be found using expression (4.5).

$$\mathrm{ADC}_{\mathrm{bit}}^{\mathrm{voltage}} = \frac{\mathrm{ADC} - \mathrm{fullscalevoltage}}{4095} \tag{4.5a}$$

$$\mathrm{ADC}_{\mathrm{bit}}^{\mathrm{current}} = \frac{\mathrm{ADC} - \mathrm{fullscalecurrent}}{4095} \tag{4.5b}$$

For the GANINV module used in this example, the $\mathrm{ADC}_{\mathrm{bit}}^{\mathrm{voltage}} = 19.9\,\mathrm{mV/bit}$ and $\mathrm{ADC}_{\mathrm{bit}}^{\mathrm{current}} = 8\,\mathrm{mA/bit}$. Consequently, when using a 24 V DC bus only $\approx 1/3$ of the ADC (for voltage) bits are used. Likewise, if current amplitudes in the order of 5 A are anticipated then only $\approx 1/3$ (for peak-to peak current amplitude) is used. Hence the ADC-bit-voltage/current values could be improved for this example by tuning the hardware ADC-fullscale voltage/current settings, i.e using lower values in this case.

The current transducer voltages `v_A_LEM`, `v_B_LEM` and `v_C_LEM`, are sampled by the `Iscaling` module shown in figure 4.9. When dealing with the actual MCU, the ADC converter also takes over this sampling process. Apart from the measured voltage/currents generated by the converter, the PSIM coupler unit also provides the actual motor model current vector: `i_alp`, `i_bet` (in stationary coordinates), phase 'A' voltage and speed, which are represented by the variables `v_A`, `i_alp`,`i_bet` and `n_m` (after conversion to rpm) respectively (see figure 4.9).

Results achieved with the COSIM approach are shown in figure 4.13.
Indicated in this figure is the phase 'A' voltage waveform together with the reference voltage, phase current, shaft torque and shaft speed.
A vector plot, as given in figure 4.14, at time mark 0.069s after starting the co-simulation, shows the measured LPF voltage vector ('blue'), unity d-axis vector ('green') and the actual (from the IRTF motor model) current vector ('red'). Note that the impact of switching is clearly reflected in the current vector locus, e.g. changes in end-point direction due to use of 'active' or 'zero' voltage vectors [13]. Co-simulation, as discussed in this sub-section, is instructive for understanding the impact of using a switching converter on drive operation.

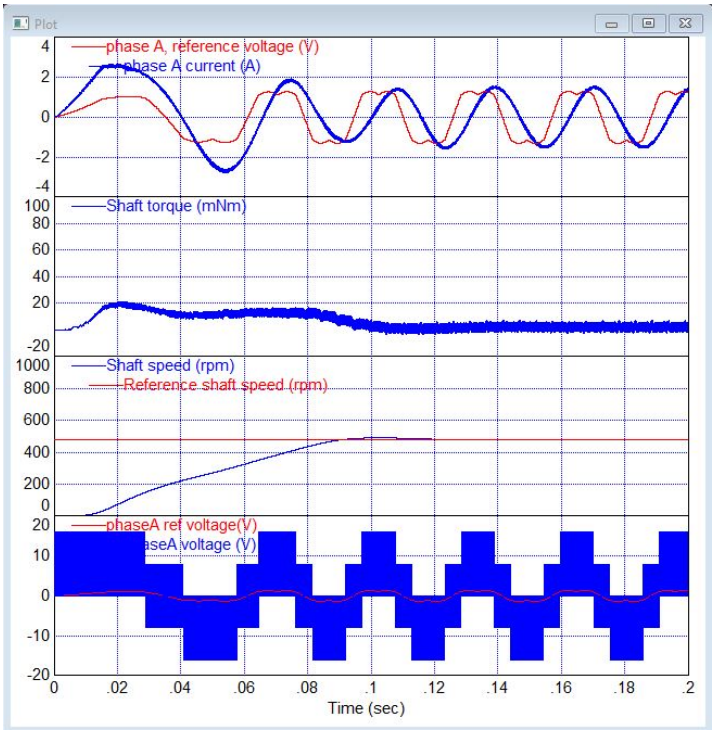


Fig. 4.13 COSIM result, voltage & frequency control PMsyn drive

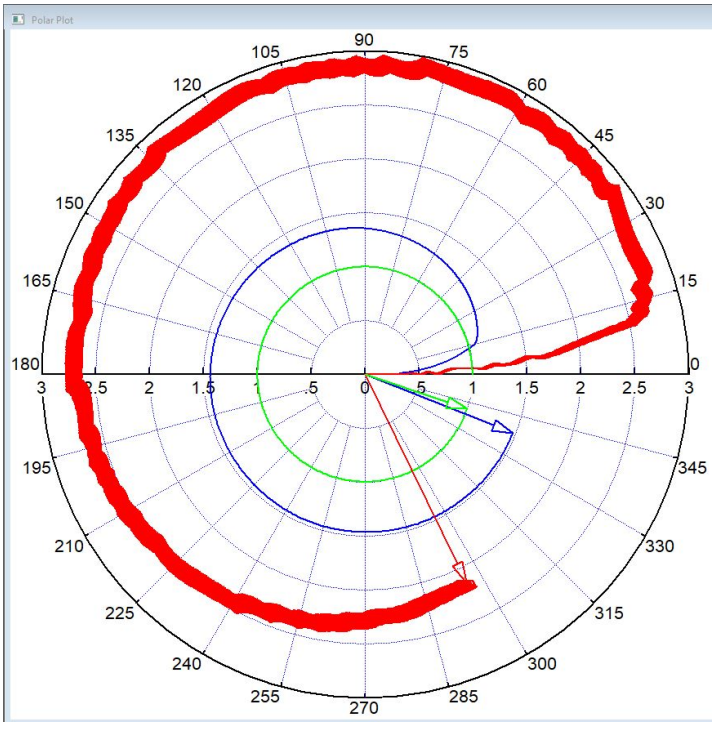


Fig. 4.14 COSIM result, voltage & frequency control PMsyn drive: vector plot

4.1.1.4 Voltage & Frequency control-F0049-GANINV: HIL

Reference files:

- `V_F280049-GANINV-compile.vsm`, for HIL-compile
- `V_F280049-GANINV-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.134 represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F280049` module) which is connected to a PMsyn Teknic motor[2].

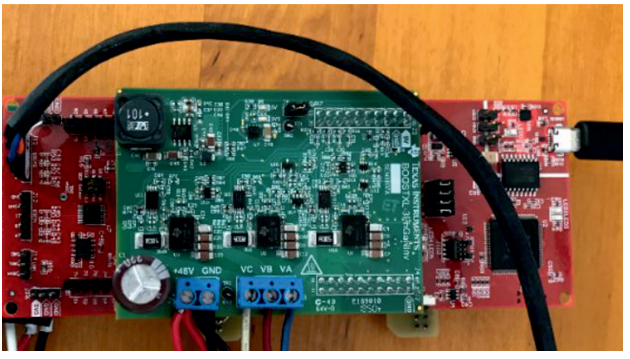


Fig. 4.15 MCU-converter setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.16 shows the 'voltage controller' module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct/quadrature reference voltages `ud`, `uq` and the frequency `Freq`, which are converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Further information on this tool will be given at a later stage. Outputs of the Controller module are the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts. In addition, a `CPU` output is added, which provides the user with a % reading of MCU activity, relative the controller sample time.

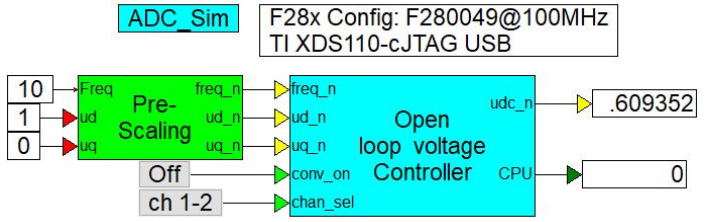


Fig. 4.16 HIL-Compile model of a PMsyn drive with voltage & frequency control

Also shown in figure 4.16 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 4.17.

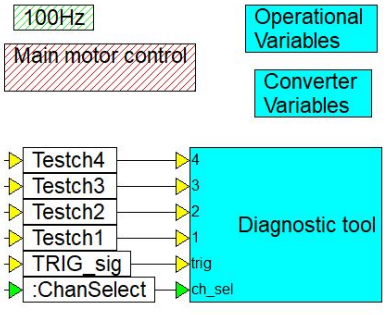


Fig. 4.17 One level down into the 'Open loop Voltage Controller' module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.18), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage/frequency.
figure 4.18.
The 'Converter Variables' dialog box (see figure 4.19), defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.4).
A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A

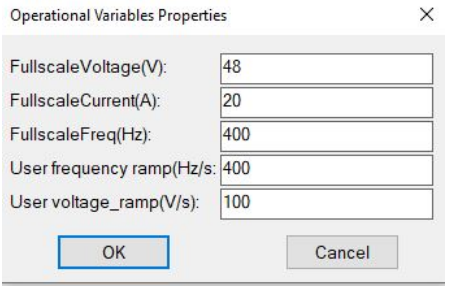


Fig. 4.18 HIL: Dialogue box 'Operational Variables'

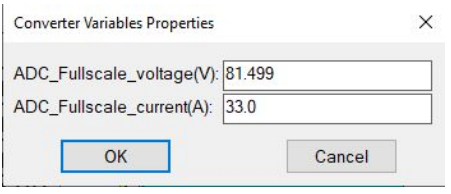


Fig. 4.19 HIL: Dialogue box 'Converter Variables'

'two channel' multiplexer is used to allow the user to either display variables `Test1`, `Test2` or `Test3`, `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.17 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 4.20. Of these, the 'Voltage controller' has been discussed previously (see figure 4.3), where it was used in sections: SIM, PIL and COSIM of the respective development tool-chain.

The per unit variables: DC bus voltage, half-bridge currents and voltages, are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs

- Board in use: `LAUNCHXL-F280049` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after every second (in this case) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

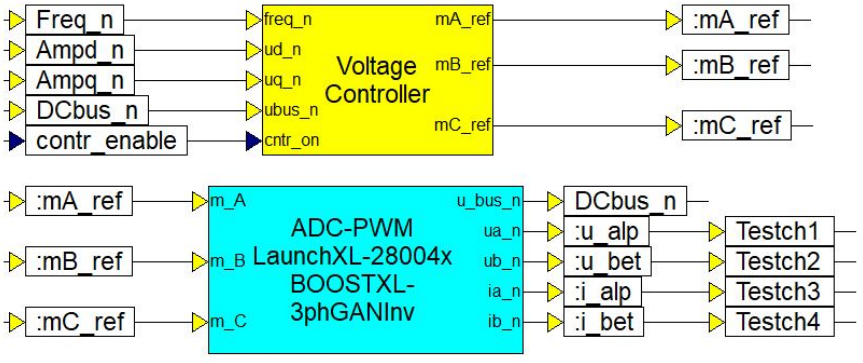


Fig. 4.20 One level down into the 'Main motor controller' module

Moving one level down the `ADC-PWM` unit (given in figure 4.20), shows the `PWM unit BOOSTXL-GANINV` module, that controls the converter switches based on the modulation indices generated by the controller (see figure 4.21).

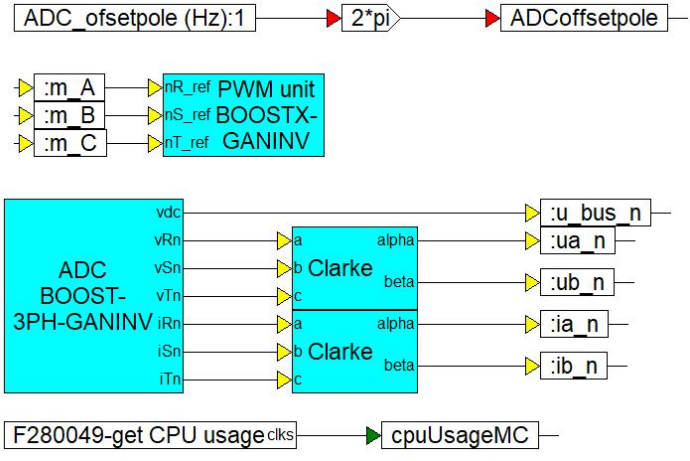


Fig. 4.21 One level down into the 'ADC-PWM' module

The `ADC BOOSTXL 3PH-GANINV` unit is used to measure the three low-pass filtered half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `ua_n`, `ub_n` using a forward Clarke conversion. Similarly, the three measured phase currents are converted to stationary coordinates α, β with outputs `ia_n`, `ib_n` respectively.
The ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage, `u_bus_n` completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example (see figure 4.22), and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (discussed in the previous section).

Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct/quadrature voltage reference values and the electrical frequency. Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

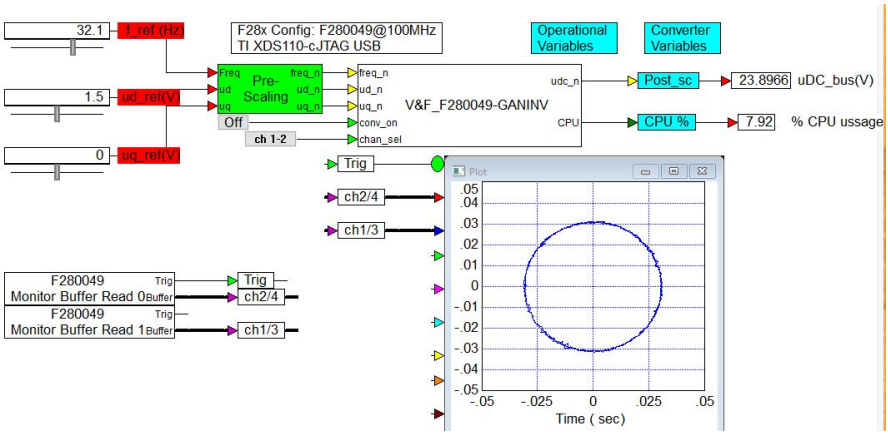


Fig. 4.22 HIL-Debug model, F0049-GANINV of a PMsyn drive with voltage & frequency control

The plot given in figure 4.22 displays the per unit low-pass filtered (LPF) measured voltage vector locus. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus. Note that the voltage reference slider `ud_ref` is set to 1.5V (and `uq_ref`=0),

hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{1.5}/_{48} = 0.031$, given that a full-scale voltage of u_{fs}=48 V is used. Observation of the plot result confirms that the amplitude is approximately equal to the anticipated value. Some amplitude deviation will inevitably occur due to practical issues such as converter dead-time and voltage drop across the converter switches. The electrical frequency reference in figure 4.22 is set to 32.1Hz, which corresponds to a shaft speed equal to $n_m = {}^{60\,f_{ref}}/_{p} = 481$ rpm, given that a four pole pair machine is used, hence $p = 4$.

Post analysis using Voltage and freq control

A key issue that needs to addressed for current/frequency control (to be discussed a later stage) is the orientation of the current vector relative to the applied voltage vector. Said current vector must LAG the voltage vector. For this purpose, the HIL 'V&F' drive can be used effectively, as will be discussed in this paragraph. If this orientation is NOT correct, then there are issues with the ADC voltage/current signal conditioning that need to resolved in the HIL drive. Note that when operating under no-load conditions (as is the case in this HIL example) the rotor flux and current will be aligned (no torque produced) as shown in figure 4.23.

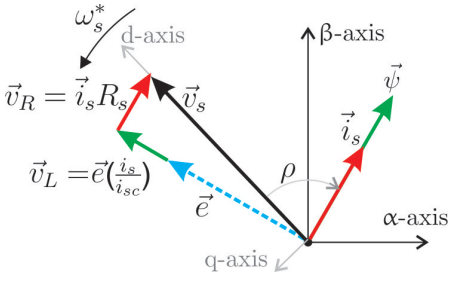


Fig. 4.23 HIL-Debug model, orientation of the voltage/current vectors-no-load operation

Observation of this plot confirms that the current vector lags the voltage vector (we are assuming anti-clockwise operation) and the lag angle ρ can be calculated using:

$$\rho = \arctan\left(\frac{e}{i_s R_s}\left(1 + \frac{i_s}{i_{\text{sc}}}\right)\right) \tag{4.6}$$

where $e = 2\pi f_s^* \psi$ and $i_{\text{sc}} = {}^{\psi}/_{L_s}$ represents the EMF (that corresponds with the set operating frequency) and short-circuit current of the machine

respectively. In this HIL example (see figure 4.22), the reference frequency was set to 32.1 Hz and the current vector amplitude i_s was found to be 0.78 A (determined from the current vector locus amplitude in the plot by selecting `ch 3-4` in figure 4.22). Consequently, the expected lag angle can be calculated using equation (4.6), which gives $\rho = 76\,^\circ$. Note that the short-circuit current for the PM machine is large compared to the current amplitude (in this example), hence the lag angle can in this case be found using $\rho \approx \arctan\left({}^{e}/_{i_s R_s}\right)$.

To check the orientation and amplitude of the current vector relative to the voltage vector it is prudent to look at the α, β components. This can be by done by moving the `TEST` signals shown in figure 4.20 to : `Testch1=u_alp`, `Testch2=i_alp`, `Testch3=u_bet` and `Testch4=i_beta`. Running the HIL (after recompiling to generate a new xxx.out file) gives the result shown in figure 4.24 (`XY plot` disabled in the HIL debug plot module), with channel select button `ch 1-2` active. Selecting `ch 3-4` will shown the per unit u_β, i_β signals.

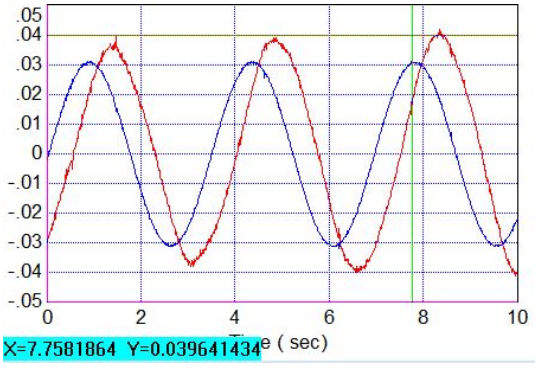


Fig. 4.24 HIL-Debug model, per unit u_α, i_α waveforms-no-load operation

Observation of figure 4.24, confirms that the current ('red' trace) is indeed lagging the voltage waveform and the time delay between these two is in agreement with the calculated ρ value.

For comparative purposes an oscilloscope with DC current probe was used to measure the phase 'A' current, as shown in figure 4.25.
The measured per unit current amplitude (after scaling by the full-scale current of 20 A) is in agreement with the current shown in figure 4.24, which has the same voltage/frequency reference settings as used in this HIL example. Hence, this post analysis confirms that the ADC full-scale current value for this converter is indeed correct. Furthermore, the polarity of the current waveforms have been checked to avoid problems when trying to implement current control.

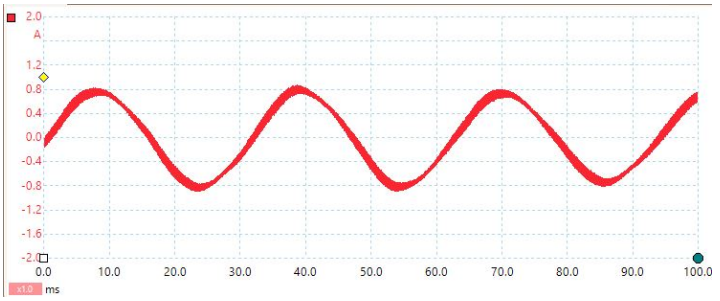


Fig. 4.25 experimental results: measured phase current, phase 'A'

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.1.1.5 Voltage & Frequency control-F069-GANINV: HIL

Reference files:

- `V_F28069-GANINV-compile.vsm`, for HIL-compile

- `V_F28069-GANINV-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.26 represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F28069` module) which is connected to a PMsyn Teknic motor[2].

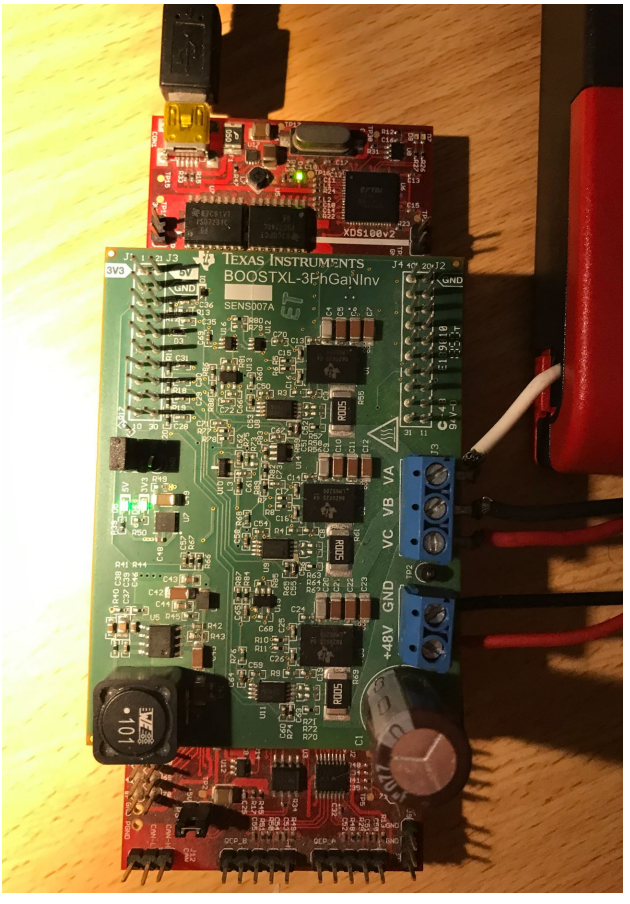


Fig. 4.26 MCU-converter setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of the HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller

configuration. The model given in figure 4.27 shows the 'open loop voltage controller' module, which must be compiled to generate an out-file.

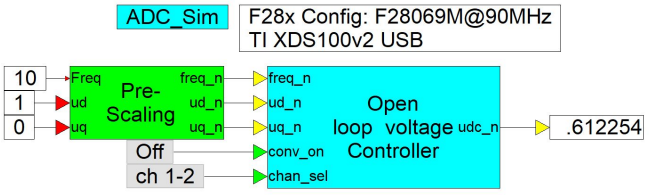


Fig. 4.27 HIL-Compile model of a PMsyn drive with voltage & frequency control

Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct/quadrature reference voltages `ud`, `uq` and the frequency `Freq`, which are converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Output of the Controller module is the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
Also shown in figure 4.27 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 4.28.

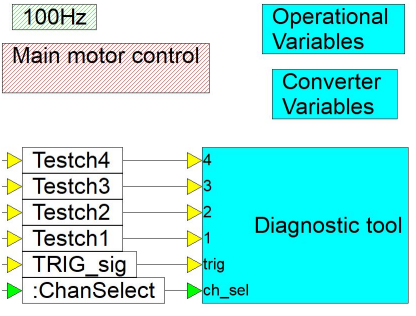


Fig. 4.28 One level down into the 'Open loop Voltage Controller' module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.29), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage/frequency.

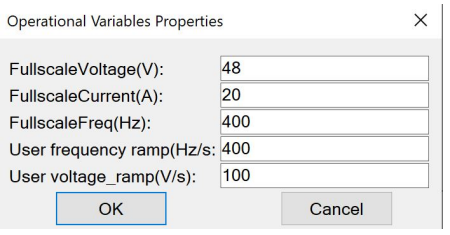


Fig. 4.29 HIL: Dialogue box 'Operational Variables'

The 'Converter Variables' dialog box (see figure 4.30), defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.4).

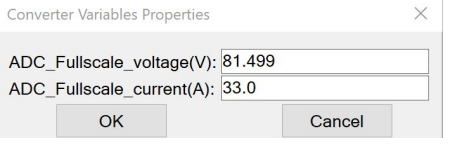


Fig. 4.30 HIL: Dialogue box 'Converter Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in the HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1`, `Test2` or `Test3`, `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.28 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 4.31. Of these, the 'Voltage controller' has been discussed previously, for a different MCU, see figure 4.3, where it was used in phases : SIM, PIL and COSIM of the respective development tool-chain.

The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 30 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F28069` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed every second (in this case) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

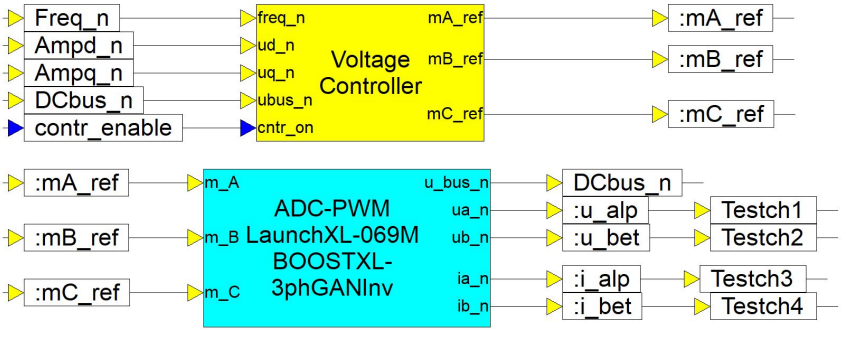


Fig. 4.31 One level down into the 'Main motor controller' module

Moving one level down in the `ADC-PWM` unit (given in figure 4.31), shows a `PWM unit BOOSTXL-GANINV`, that controls the converter switches based on the modulation indices generated by the controller (see figure 4.32).
The `ADC BOOSTXL 3PH-GANINV` unit is used to measure the three low-pass filtered half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp_n`, `u_bet_n` using a forward Clarke conversion. Similarly, the three measured phase currents are converted to stationary coordinates α, β with outputs `i_alp_n`, `i_bet_n` respectively.

ADC offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `u_DCbus_n`, completes the ADC measurement sequence.

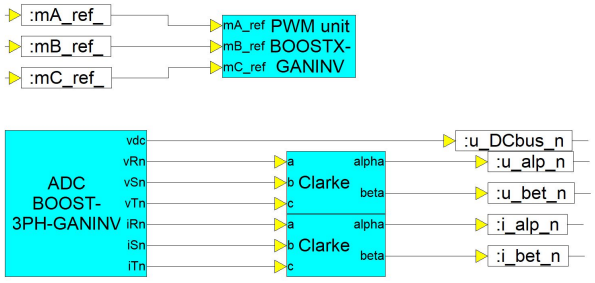


Fig. 4.32 One level down into the 'ADC-PWM' module

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 4.33 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).

Central to this figure is the Embed 'target execution module', which executes the out-file. Three sliders are used to set the direct/quadrature voltage reference values and the electrical frequency. Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.
The plot given in figure 4.33 displays the per unit LPF measured voltage vector locus. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus. Note that the voltage reference slider `ud_ref` is set to 1.5V (and `uq_ref`=0), hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{1.38}/_{48} = 0.028$, given that a full-scale voltage of u_{fs}=48 V is used. Observation of the plot result confirms that the amplitude is approximately equal to the anticipated value. Some amplitude deviation will inevitably occur due to practical issues such as: converter dead-time and voltage drop across the converter switches. The electrical frequency reference in figure 4.33 is set to 22 Hz, which corresponds to a shaft speed equal to $n_m = {}^{60\, f_{ref}}/_{p} = 330$ rpm, given that a four pole pair machine is used, hence $p = 4$.

For comparative purposes an oscilloscope with a DC current probe was used to measure the phase 'A' current, shown in figure 4.34. The current amplitude shown in this figure, matches the per unit current locus amplitude (when selecting `ch 3-3` for the scope module, which confirms that the `ADC_fullscale_current` setting is correct.
Prior to activating this type of laboratory, the reader needs to be aware of

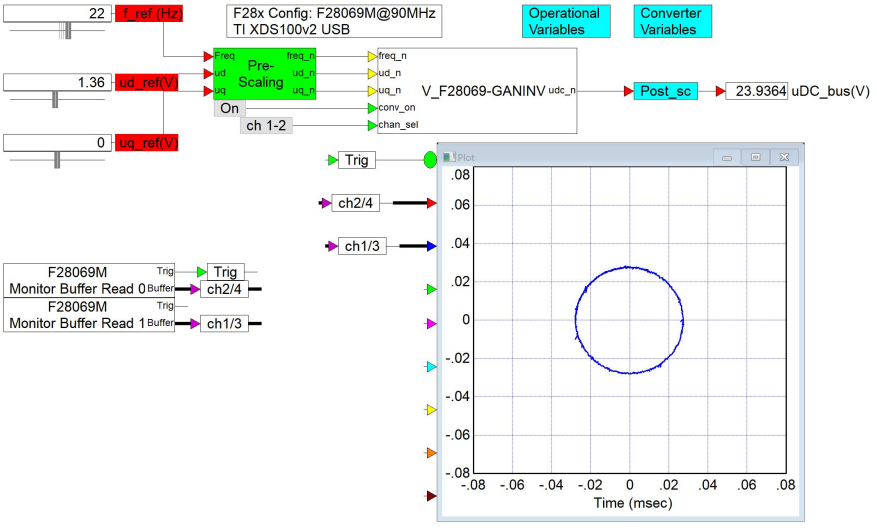


Fig. 4.33 HIL-Debug model, F069-GANINV of a PMsyn drive with voltage & frequency control

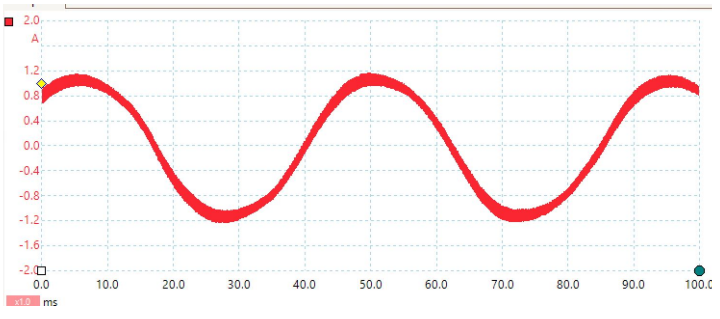


Fig. 4.34 experimental results: measured phase current, phase 'A'

the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).

- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the numeric display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.1.1.6 Voltage & Frequency control-F027-boostXL8301: HIL

Reference files:

- `V&F_027__boostXL_8301-compile.vsm`, for HIL-compile
- `V&F_027__boostXL_8301-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), shown in figure 4.35, represents the `BOOSTXL-8301` converter (located on the `LAUNCHXL-F28027` module) which is connected to a PMsyn Teknic motor[2].

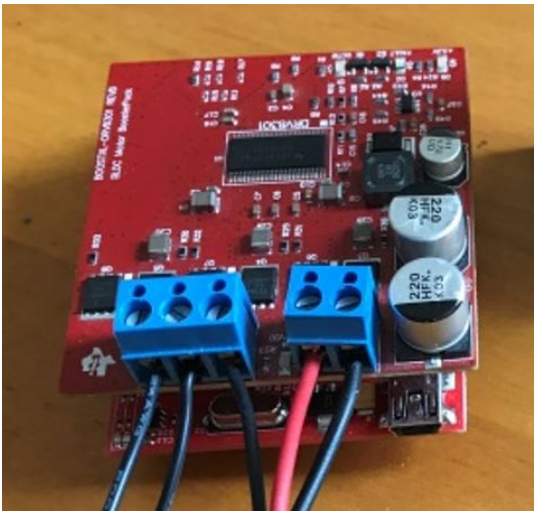


Fig. 4.35 F28027 MCU-boostXL8301 converter setup for HIL operation

For HIL operation, the drive board itself needs to be known in considerable detail. Several aspects need to be considered such as, allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage is used to generate the file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.36 shows the 'voltage controller' module, which must be compiled to generate an out-file.

Note that for this HIL, the out-file that is compiled, MUST be located in the flash memory of the MCU. This can be done with the aid of the software 'Uniflash' [30].

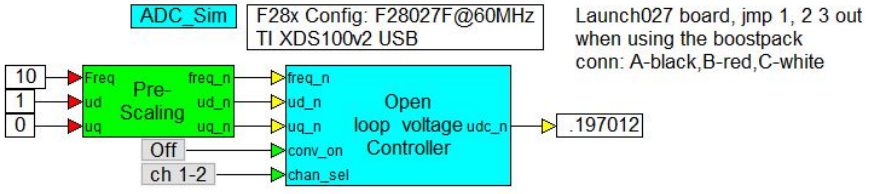


Fig. 4.36 HIL-Compile model of a brushed DC drive with voltage & frequency control

Inputs, shown in this stage as constants (for readability purposes) for this controller module, are the user direct and quadrature reference voltages `ud`, `uq` and the frequency `Freq`, which are converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels to be viewed. Outputs of the Controller module are the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.

Also shown in figure 4.36 is an 'ADC-Sim' module, which is used to provide artificial inputs (as pre-set in this module) to the ADC channels. Hence these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes, as shown in figure 4.37.
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue

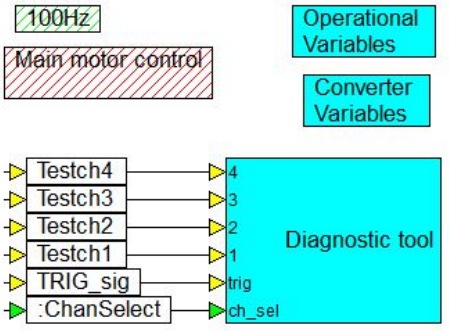


Fig. 4.37 One level down into the 'Open loop Voltage Controller' module

box (see figure 4.38), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage/frequency.

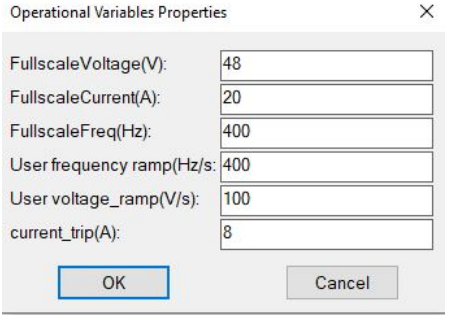


Fig. 4.38 HIL: Dialogue box 'Operational Variables'

The 'Converter Variables' dialogue box (see figure 4.39), defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.4).

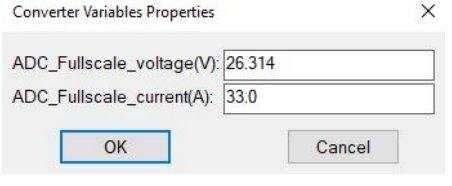


Fig. 4.39 HIL: Dialogue box 'Converter Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently be displayed in a graph or plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using the 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.37 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit.
Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 4.40. Of these, the 'Voltage controller' has been discussed previously, for a different MCU (see figure 4.3), where it was used in sections: SIM, PIL and COSIM of the respective development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 15 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: 0.16 μs
- Board in use: `LAUNCHXL-F28027` with `BOOSTXL-8301` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.
The `ADC-PWM Launch027 BOOST` unit is used to generate the PWM and measure the three (per unit) low-pass filtered half-bridge voltages which are then converted to stationary coordinates α, β voltages: `u_alp_n`, `u_bet` using a forward Clarke conversion. Similarly, the three (per unit) measured phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively.

ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by a '100Hz' module, as mentioned previously. The per unit DC bus voltage, `DCbus_n` completes the ADC measurement sequence.

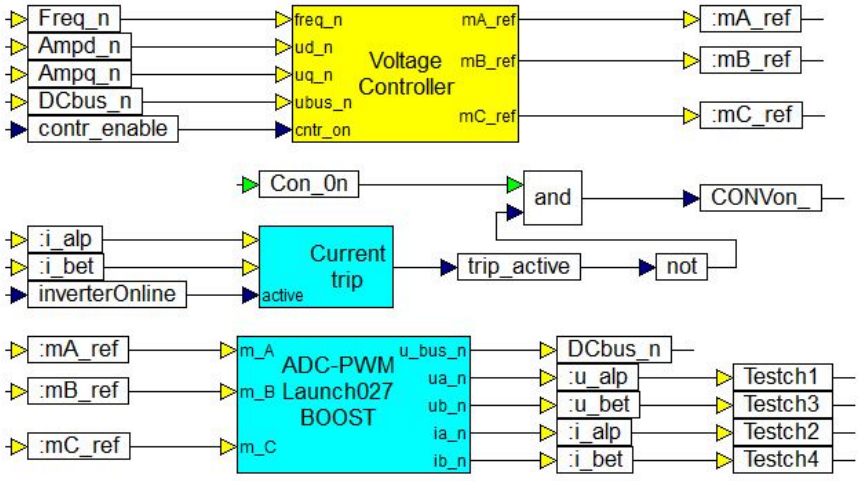


Fig. 4.40 One level down into the 'Main motor controller' module

The HIL-debug/run tool-chain component is the operational phase of this example, shown in figure 4.41 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above). As mentioned above, said out-file must be loaded to MCU flash memory using, for example, the software 'UniFlash'.

Central to this figure is the Embed 'target execution module', which executes the out-file. Three sliders are used to set the direct/quadrature voltage reference values and the electrical frequency. A post-scaling module is used to convert the per-unit measured DC bus voltage to actual value, as shown, with a numeric display. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals. The plot given in figure 4.41 displays the per unit LPF measured voltage `u_alp` ('blue') and the per unit current `i_alp`. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current and LPF voltage for the β components (see figure 4.40). Note that the voltage reference slider `ud_ref` is set to 1.5 V (and `uq_ref`=0), hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{1.5}/_{48} = 0.031$, given that a full-scale voltage of u_{fs}=48 V is used. Observation of the plot, confirms that the voltage amplitude (shown in 'blue') is approximately equal to the anticipated value. Some amplitude deviation will inevitably occur due to practical issues such as: converter dead-time and voltage drop across the converter switches. The electrical frequency reference in figure 4.41 is set to 25 Hz, which corresponds to a shaft speed equal to $n_m = {}^{60\, f_{ref}}/_{p} = 375$ rpm, given that a four pole pair machine is used, hence $p = 4$.

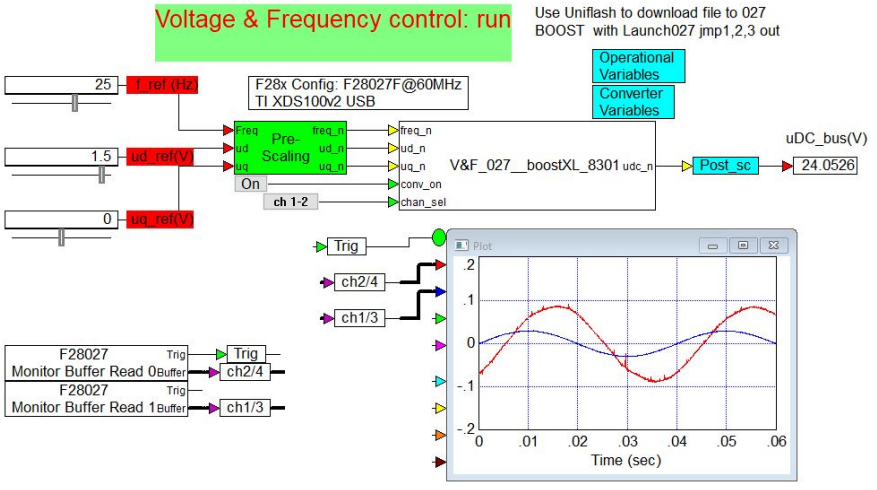


Fig. 4.41 HIL-Debug model, F027-boostXL8301 of a PMsyn drive with voltage & frequency control

The per unit phase current amplitude is equal to 0.084, which corresponds to a real value of 1.68 A. It is prudent to check (for both α, β components) that the current (shown in 'red') LAGS the voltage), given the presence of an inductive load, which is indeed the case (see figure 4.41).

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.1.1.7 Voltage & Frequency control-F069-DRV8301HC: HIL

Reference files:

- `V&F_069_8301_HC-compile.vsm`, for HIL-compile
- `V&F_069_8301_HC-run_d.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.42 represents the `DRV8301-HC` converter (linked to the `LAUNCHXL-F28069M` module) which is connected to a PMsyn Teknic motor[2].

Fig. 4.42 F28069 MCU-8301HC converter setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects need to be considered such as allocation of data acquisition channels relative to the converter voltage and current sensors, choice of PWM

outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.43 shows the 'voltage controller' module, which must be compiled (by selecting this module) to generate an out-file.

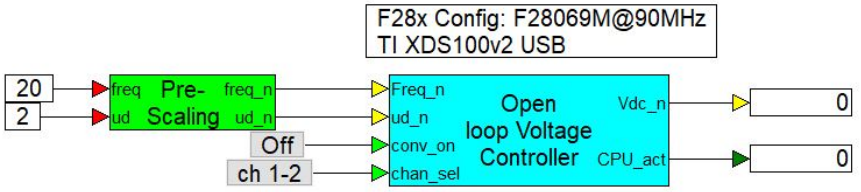


Fig. 4.43 HIL-Compile model of a brushed DC drive with voltage & frequency control

Inputs, shown in this stage as constants (for readability purposes) for this controller module, are the user direct axis reference voltage `ud` (quadrature reference `uq` set to zero) and the frequency `freq`, which are converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Outputs of the Controller module are the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts, and CPU activity monitor `CPU_act`.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes, as shown in figure 4.44. Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'operational Variables' dialogue box, see figure 4.45, is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage/frequency. The 'Converter Variables' dialogue box, as given in figure 4.46, defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.4).
A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

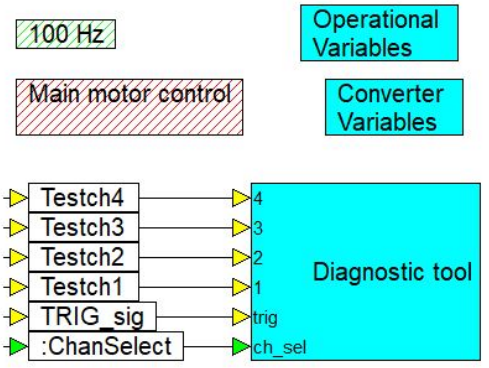


Fig. 4.44 One level down into the 'Open loop Voltage Controller' module

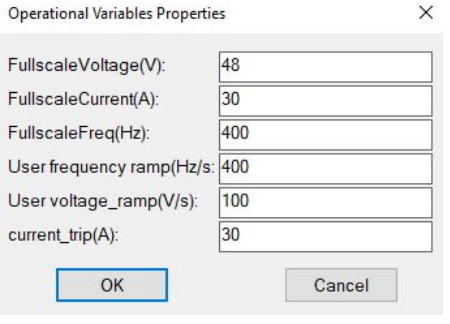


Fig. 4.45 HIL: Dialogue box 'Operational Variables'

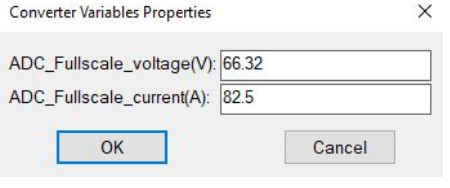


Fig. 4.46 HIL: Dialogue box 'Converter Variables'

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.44 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end

of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 4.40. Of these, the 'Voltage controller' has been discussed previously, for a different MCU (see figure 4.3), where it was used in sections: SIM, PIL and COSIM of the respective development tool-chain.

The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 15 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: 0.1 μs
- Board in use: `F28069` ISO control card with `DRV-8301HC` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

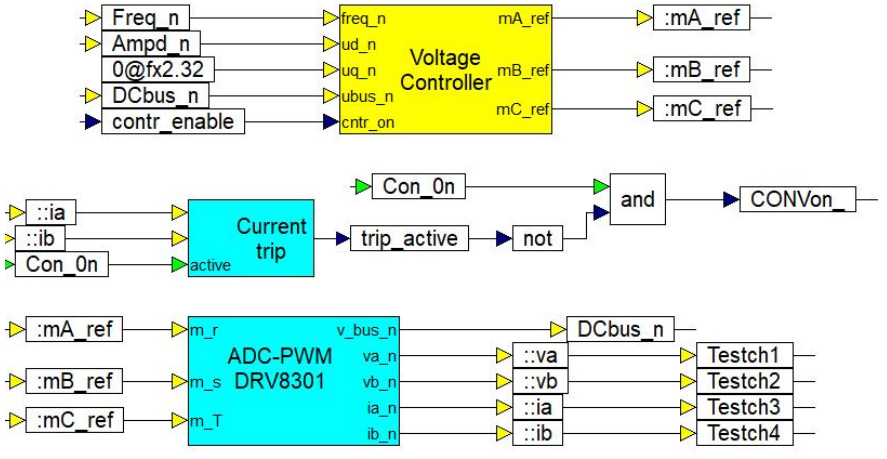


Fig. 4.47 One level down into the 'Main motor controller' module

The `ADC-PWM DRV8301` unit is used to generate the PWM and measure the three (per unit) low-pass filtered half-bridge voltages, which are then converted to stationary coordinates α, β voltages `v_a`, `v_b` using a forward Clarke conversion. Similarly, the three (per unit) measured phase currents are converted to stationary coordinates α, β with outputs `i_a`, `i_b` respectively.
The ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offsets is controlled by

a '100Hz' module, as mentioned previously. The per unit DC bus voltage `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational phase of this example (see figure 4.48), and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above). Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct/quadrature voltage reference values and the electrical frequency. A post-scaling module is used to convert the per-unit measured DC bus voltage to actual value, as shown with a numeric display. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

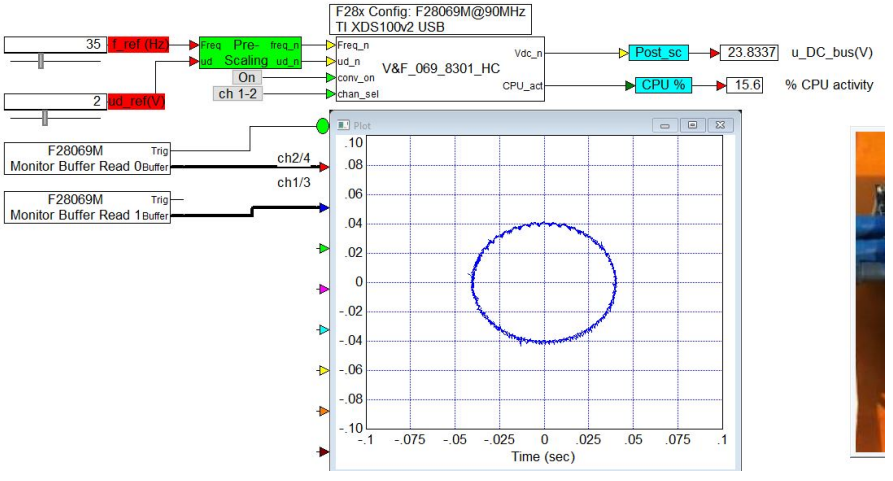


Fig. 4.48 HIL-Debug model, F28069-DRVXL8301HC of a PMsyn drive with voltage & frequency control

The plot given in figure 4.48, displays the per unit LPF, measured voltage vector `v_a`, `v_b`. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus (see figure 4.47). Note that the voltage reference slider `ud_ref` is set to 2.0 V (and `uq_ref`=0), hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{2.0}/_{48} = 0.041$, given that a full-scale voltage of u_{fs}=48 V is used. Observation of the plot result confirms that the voltage amplitude is approximately equal to the anticipated value. Some amplitude deviation will inevitably occur due to practical issues such as the converter dead-time and voltage drop across the converter switches.
The electrical frequency reference in figure 4.48 is set to 35 Hz, which corresponds to a shaft speed equal to $n_m = {}^{60\, f_{ref}}/_{p} = 525$ rpm, given that a four pole pair machine is used, hence $p = 4$. The per unit phase current amplitude

is equal to 0.08, which corresponds to a real value of 2.4 A.

For comparative purposes an oscilloscope with DC current probe was used to measure the phase 'A' current, as shown in figure 4.49.

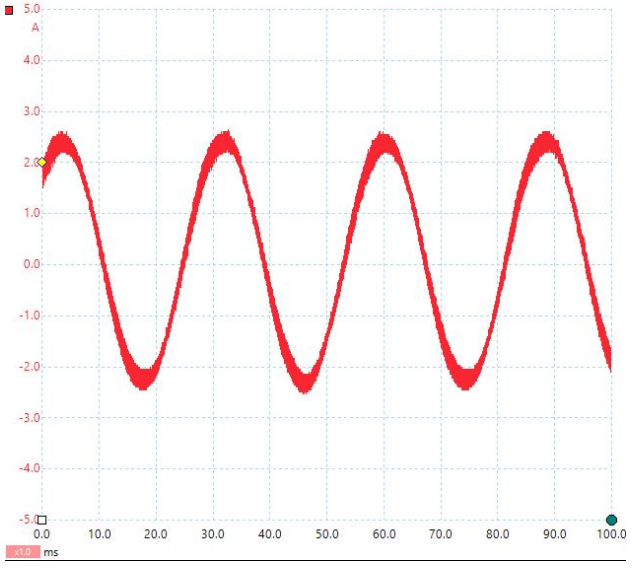


Fig. 4.49 experimental results: measured phase current, phase 'A'

The measured current amplitude is in agreement with the current vector plot, that can be found when switching to `ch 3-4` channel select. This shows the per unit locus amplitude and must therefore be multiplied with the full-scale current value of 30 A.
Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter

- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.1.1.8 Voltage & Frequency control-F069-DRV8312: HIL

Reference files:

- `V&F_F069-8312-compile.vsm`, for HIL-compile
- `V&F_F069-8312-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), shown in figure 4.50, represents the `DRV8312` converter (linked to the `LAUNCHXL-F28069M` ISO card) which is connected to a PMsyn Teknic motor[2]. For HIL operation the drive board itself needs to be known in considerable detail. Aspects to be considered include allocation of data acquisition channels relative to the converter voltage and current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.51 shows the 'voltage controller' module, which must be compiled (by selecting said module) to generate an out-file.

Analogue input, shown in this stage as a constant (for readability purposes), for this controller module is the shaft speed `Nref`, which is converted to 'per unit' electrical frequency `freq_n` in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed.
Outputs of the Controller module are the per unit DC bus voltage `DCb_n`, electrical shaft speed `shft_fen`, input power estimate `pin_n` and current vector amplitude `i_vec_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts. Also present is an indicator for MCU activity, relative to the electrical controller sampling frequency `cpu_act`.

Fig. 4.50 MCU-converter setup for HIL operation

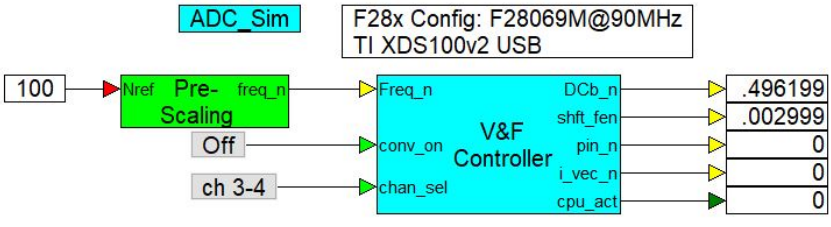


Fig. 4.51 HIL-Compile model of a PMsyn drive with voltage & frequency control

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes, as shown in figure 4.52.
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'operational Variables' dialogue box, see figure 4.53, is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference frequency. The nature of control is different compared to earlier 'voltage' controllers where both frequency AND voltage could be chosen independently. In this example the user defined reference frequency determines the modulation index directly using expression (4.7).

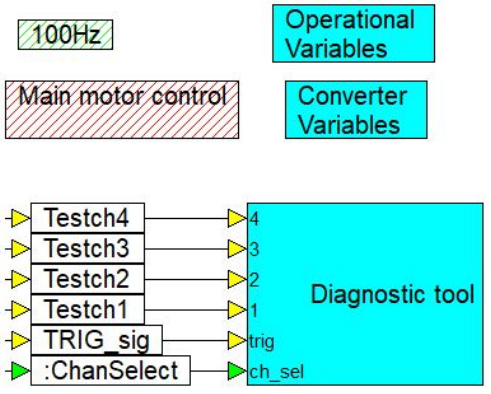


Fig. 4.52 One level down into the 'Open loop Voltage Controller' module

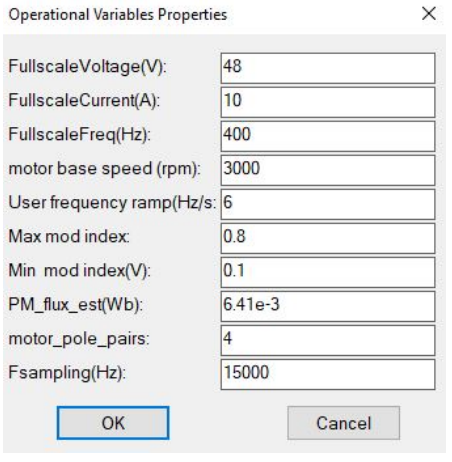


Fig. 4.53 HIL: Dialogue box 'Operational Variables'

$$m^{\text{ref}} = \frac{f_{\text{ref}}}{f_{\text{base}}} \tag{4.7}$$

where the base frequency f_{base} is defined in the dialogue box (see figure 4.53) by the 'base speed (rpm) using expression $f_{\text{base}} = \frac{n_{\text{base}}\, p}{60}$ Hz, where p is the machine pole pair number. Also shown in figure 4.53, are the maximum/minimum modulation indices, which constrains the modulation index reference defined by equation (4.7).

The 'Converter Variables' dialogue box, as given in figure 4.54, defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.4). In addition, the corner frequency of the low-pass filters for the measured converter voltages, is also added here.

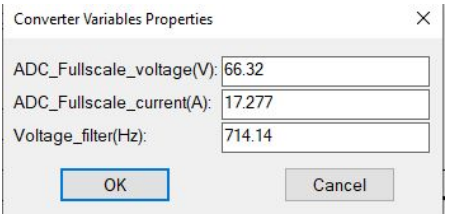


Fig. 4.54 HIL: Dialogue box 'Converter Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1`and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.52 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 4.55. Of these, the 'Voltage controller' concept used in this example, has been introduced earlier with the aid of expression (4.7). In this case the direct axis modulation index is set to $m_d = m^{\text{ref}}$ and quadrature reference index is set to zero.

The per unit variables: DC bus voltage, half-bridge currents and LPF voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 15 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: disabled
- Board in use: `F28069` ISO control card with `DRV-8312` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

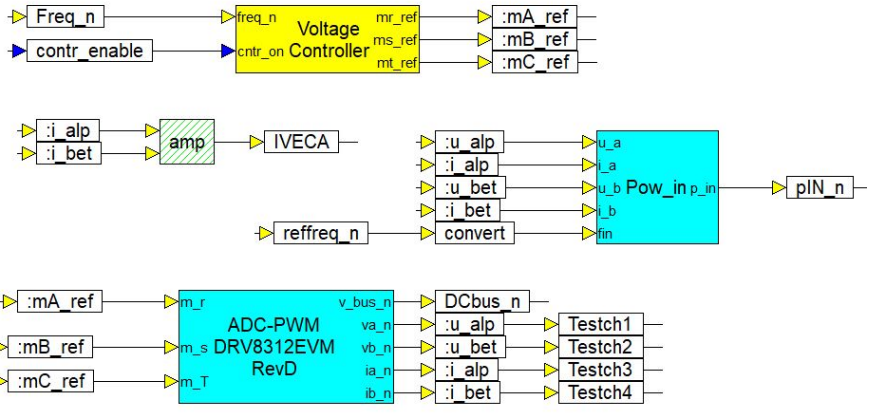


Fig. 4.55 One level down into the 'Main motor controller' module

The `ADC-PWM DRV8312EVM` unit is used to generate the PWM and measure the three (per unit) low-pass filtered half-bridge voltages which are then converted to stationary coordinates α, β voltages: `u_alp_n`, `u_bet` using a forward Clarke conversion. Similarly, the three (per unit) measured phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by a '100Hz' module, as mentioned previously.
Moving one level into the `voltage controller` module, see figure 4.56, shows the module `VwithF controller` with input the per unit reference frequency and this unit generates the (d,q) modulation indices as mentioned above.
The HIL-debug/run tool-chain component is the operational phase of this example, shown in figure 4.57 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above). Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct/quadrature voltage reference values and the electrical frequency. A post-scaling module is used to convert the per-unit measured DC bus voltage to actual value, as shown with a numeric display.
Two 'Monitor Buffer' modules are used to display two selected (using a button

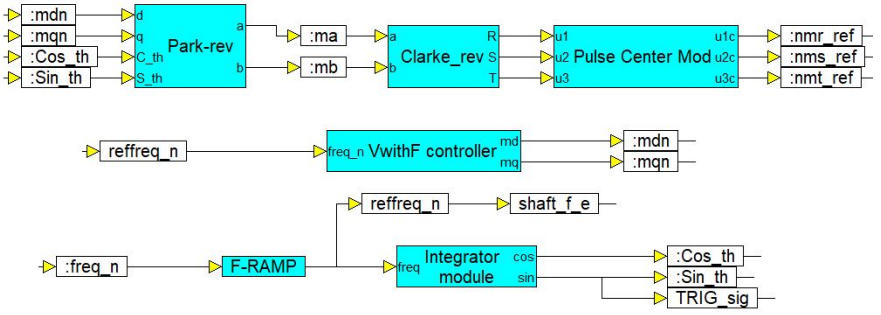


Fig. 4.56 Voltage with frequency controller for the PMsyn machine

connected to the `Ch_sel` input) diagnostic signals.

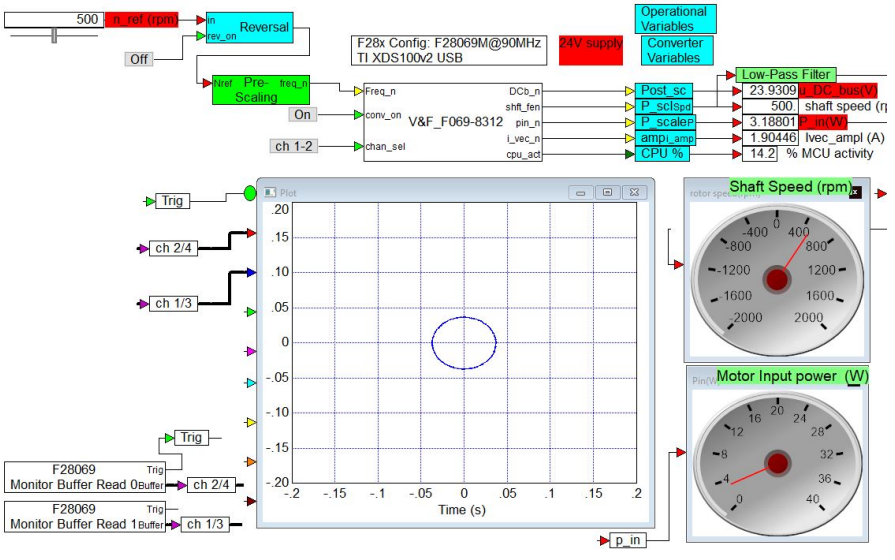


Fig. 4.57 HIL-Debug model, F28069-DRV-8312 converter of a PMsyn drive with voltage/frequency control

The plot given in figure 4.57 displays the per unit LPF measured voltage vector `u_alp`, `u_bet`. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus. Note that the speed reference slider `n_ref`, is set to 500 rpm and the selected base speed is 3000 rpm (see figure 4.53), hence the reference modulation index will be $m^{\text{ref}} = {}^{500}/_{3000} = 0.166$ (AND is within the Max/Min modulation indices selected), hence the per unit voltage amplitude should be $|\vec{u}|_n = {}^{(0.1666\times 12)}/_{48} = 0.0416$, given that a full-scale voltage of u_{fs}=48 V and

DC bus voltage of 24 V is used. Observation of the plot diagram confirms that the voltage amplitude is approximately equal to the anticipated value. Some amplitude deviation will inevitably occur due to practical issues such as converter dead-time and voltage drop across the converter switches. The numeric display shows a current vector amplitude of 1.9A, hence the per unit amplitude of the current vector locus is equal to 0.19, as may be observed by setting the channel select button to `ch 3-4`. Also shown in figure 4.57 are two gauges, which represent the shaft speed and estimated motor input power.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.1.1.9 Voltage & Frequency control-F28379-boostXL8305: HIL

Reference files:

- `V&F_F28379_boostXL8305-compile.vsm`, for HIL-compile
- `V&F_F28379_boostXL8305-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), shown in figure 4.58 represents the `BOOSTXL-8305` converter (located on the `LAUNCHXL-F28379`

module), which is connected to a PMsyn Teknic motor[2].

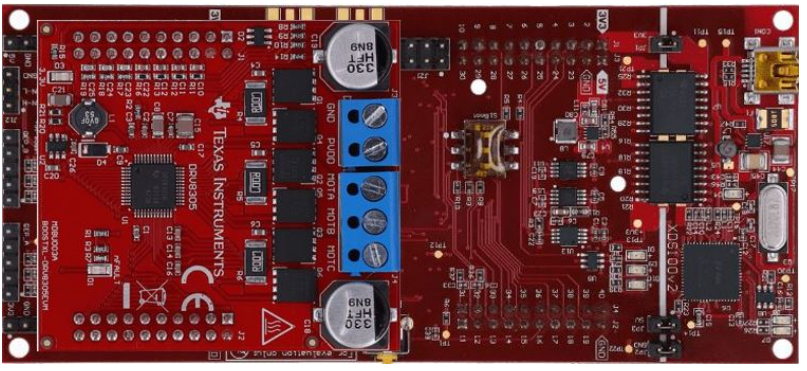


Fig. 4.58 F28379 MCU-boostXL8305 converter setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.59 shows the 'voltage controller' module, which must be compiled to generate an out-file.

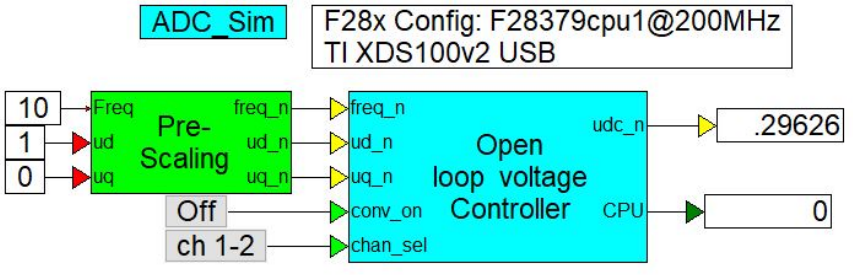


Fig. 4.59 HIL-Compile model of a PMsyn drive with voltage & frequency control

Inputs, shown in this stage as constants (for readability purposes) for this controller module are the user direct/quadrature reference voltages `ud`, `uq` and the frequency `Freq`, which are converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the

converter and to select which combination of diagnostic channels should be viewed. Outputs of the Controller module are the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts and the variable `CPU` which gives an indication of MCU usage relative to the sampling time.

Also shown in figure 4.59 is an 'ADC-Sim' module, which is used to provide artificial inputs (as pre-set in this module) to the ADC channels. Hence these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes, as shown in figure 4.60.

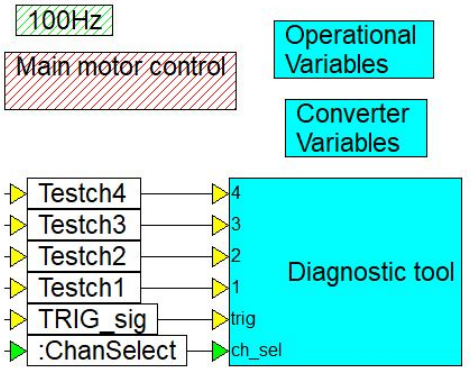


Fig. 4.60 One level down into the 'Open loop Voltage Controller' module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.61), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage-frequency.

The 'Converter Variables' dialogue box (see figure 4.62), defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.4).
A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

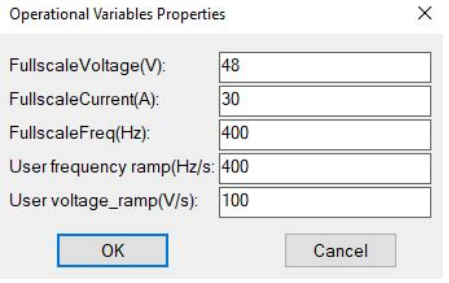


Fig. 4.61 HIL: Dialogue box 'Operational Variables'

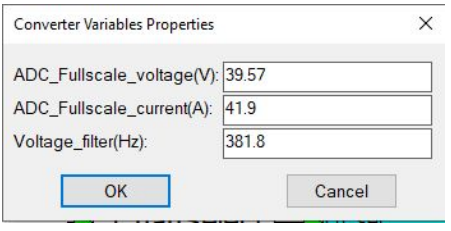


Fig. 4.62 HIL: Dialogue box 'Converter Variables'

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.60 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module, reveals a set of modules, as shown in figure 4.63. Of these, the 'Voltage controller' has been discussed previously, for a different MCU, see figure 4.3, where it was used in phases : SIM, PIL and COSIM of the respective development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 20 kHz
- Converter dead-time setting: 50 ns
- Board in use: `LAUNCHXL-F28379D` with `BOOSTXL-8305` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed

after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

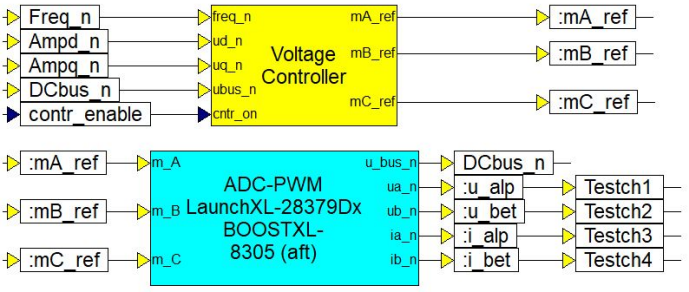


Fig. 4.63 One level down into the 'Main motor controller' module

The `ADC-PWM LaunchXL-28379D BOOSTXL8305(aft)` unit is used to generate the PWM and measure the three (per unit) low-pass filtered half-bridge voltages which are then converted to stationary coordinates α, β voltages: `u_alp_n`, `u_bet` using a forward Clarke conversion. Similarly, the three (per unit) measured phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively.

ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by a '100Hz' module, as mentioned previously. The per unit DC bus voltage `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational phase of this example, shown in figure 4.64 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above). Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct/quadrature voltage reference values and the electrical frequency. A post-scaling module is used to convert the per-unit measured DC bus voltage to actual value, as shown by a numeric display.
Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.
The plot given in figure 4.64 displays the per unit LPF measured voltage `u_alp` ('blue') and the per unit current `i_alp`. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current and LPF voltage for the β components. Note that the voltage reference slider `ud_ref` is set to 2.0 V (and `uq_ref`=0), hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{2.0}/_{48} = 0.041$, given that a full-scale voltage of u_{fs}=48 V is used.

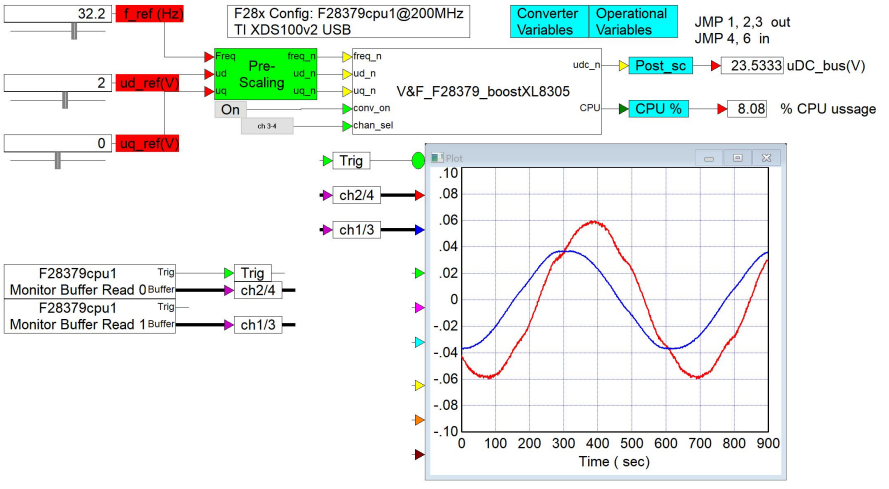


Fig. 4.64 HIL-Debug model, F283879-boostXL 8305 of a PMsyn drive with voltage & frequency control

Observation of the plot result ('blue' waveform) confirms that the voltage amplitude is approximately equal to the anticipated value. Some amplitude deviation will inevitably occur due to practical issues such as converter dead-time and voltage drop across the converter switches. The electrical frequency reference in figure 4.64 is set to 32.2 Hz, which corresponds to a shaft speed equal to $n_m = {}^{60}\, f_{ref}/p = 483$ rpm, given that a four pole pair machine is used, hence $p = 4$. The per unit phase current amplitude is equal to 0.06, which corresponds to a real value of 1.8 A. It is prudent to check (for both α, β components) that the current LAGS the voltage (otherwise current control will NOT work correctly), given the presence of an inductive load. Note that the % MCU usage time is equal to 8% of the controller sample time.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).

- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.1.1.10 Voltage & Frequency control-F069-boostXL8305: HIL

Reference files:

- `V&F_069_8305-compile.vsm`, for HIL-compile
- `V&F_069_8305-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.65 represents the `boostXL-8305` converter (linked to the `LAUNCHXL-F28069M` module) which is connected to a PMsyn Teknic motor[2]. For HIL operation

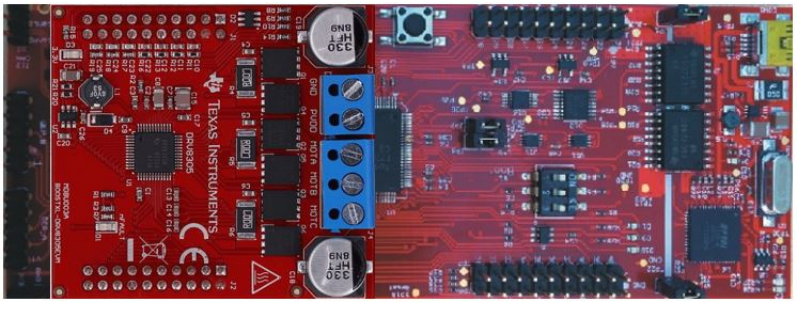


Fig. 4.65 F28069 MCU-8305 converter setup for HIL operation

the drive board itself needs to be known in considerable detail. Aspects need to be considered such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage is used to generate the file `XXX.out`, that is used in the 'debug'(run) phase

of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.66 shows the 'voltage controller' module, which must be compiled to generate an out-file.

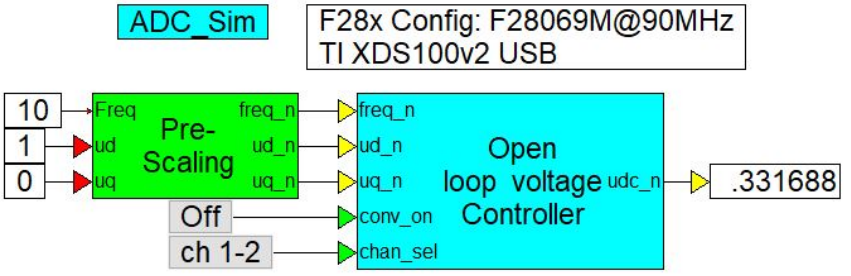


Fig. 4.66 HIL-Compile model of a PMsyn drive with voltage & frequency control

Inputs, shown in this stage as constants (for readability purposes), for this controller module are the user direct axis reference voltage `ud`, quadrature reference `uq` and frequency `freq`, which are converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Output of the Controller module is the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.

Moving one level down into the Controller module ('right' mouse click, after selecting module), leads to the set of modules and dialogue boxes, as shown in figure 4.67.
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.68), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage-frequency.
The 'Converter Variables' dialogue box, as given in figure 4.69, defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.4).
Note that a slight change to the `FullscaleVoltage` was needed, when using the `boostXL8305` board with a `LaunchXL-F28069` board. A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

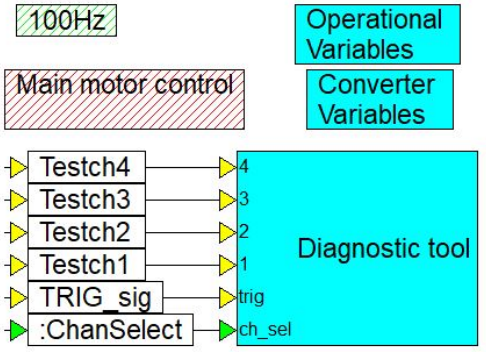


Fig. 4.67 One level down into the 'Open loop Voltage Controller' module

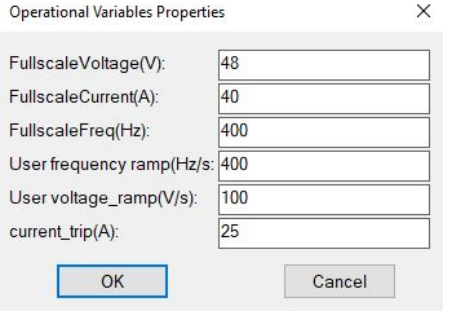


Fig. 4.68 HIL: Dialogue box 'Operational Variables'

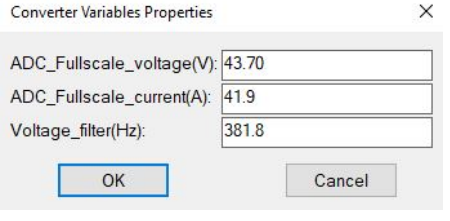


Fig. 4.69 HIL: Dialogue box 'Converter Variables'

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (State machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.67 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit

is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 4.70. Of these, the 'Voltage controller' has been discussed previously, for a different MCU (see figure 4.3), where it was used in sections: SIM, PIL and COSIM of the respective development tool-chain.

The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module.
This module is configured with the following basic settings :

- PWM frequency: 15 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: 0.1 μs
- Board in use: `F28069` ISO control card with `BOOSTXL-8305` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

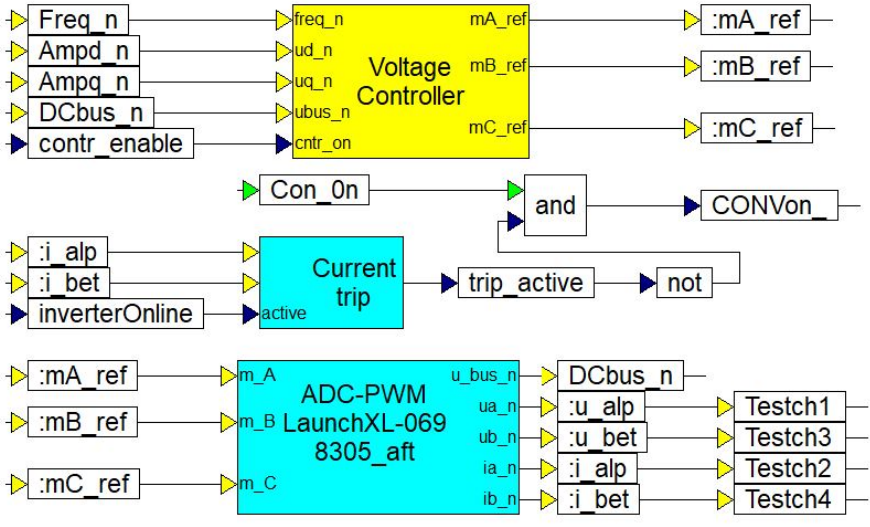


Fig. 4.70 One level down into the 'Main motor controller' module

The `ADC-PWM Launch069 8305_aft` unit is used to generate the PWM and measure the three (per unit) low-pass filtered half-bridge voltages, which are then converted to stationary coordinates α, β voltages: `u_alp_n`, `u_bet` using a forward Clarke conversion. Similarly, the three (per unit) measured phase currents are converted to stationary coordinates α, β with outputs `i_alp`,

`i_bet` respectively.

ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by a '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational phase of this example, shown in figure 4.71 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above). Central to this figure is the Embed 'target execution module', which executes said out-file.
Three sliders are used to set the direct/quadrature voltage reference values and the electrical frequency. A post-scaling module is used to convert the per-unit measured DC bus voltage to actual value, as shown with a numeric display. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals. The plot given in

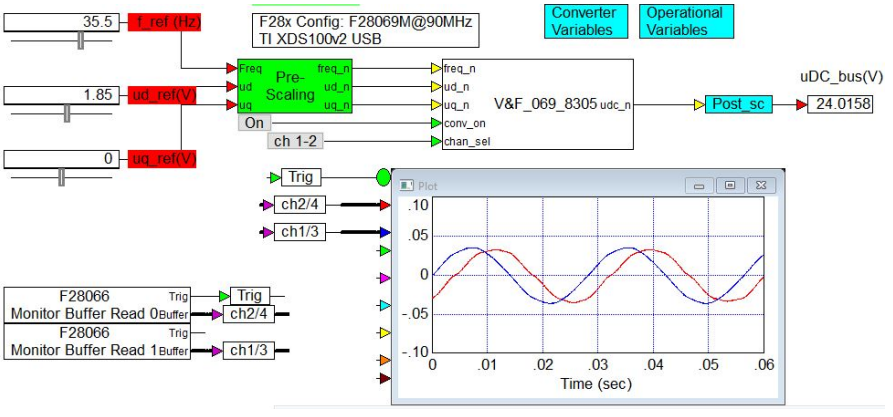


Fig. 4.71 HIL-Debug model, F28069-8305 of a PMsyn drive with voltage & frequency control

figure 4.71 displays the per unit LPF measured voltage vector `u_alp` ('blue') and per unit current `i_alp`. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit β voltage/current variables. A good check to make (for both α, β components) is that the current LAGS the voltage, given the presence of a inductive load. Such a check is essential BEFORE applying any type of current control.
Note that the voltage reference slider `ud_ref` is set to 1.85 V (and `uq_ref`=0), hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{1.85}/_{48} = 0.0385$, given that a full-scale voltage of u_{fs}=48 V is used. Observation of the plot result confirms that the voltage amplitude is approximately equal to the anticipated

value. Some amplitude deviation will inevitably occur due to practical issues such as converter dead-time and voltage drop across the converter switches. The electrical frequency reference in figure 4.71 is set to 35.5 Hz, which corresponds to a shaft speed equal to $n_m = {}^{60\,f_{ref}}/_{p} = 532$ rpm, given that a four pole pair machine is used, hence $p = 4$. The per unit phase current amplitude is equal to 0.0321, which corresponds to a real value of 1.28 A.

For comparative purposes an oscilloscope with DC current probe was used to measure the phase 'A' current, as shown in figure 4.72.

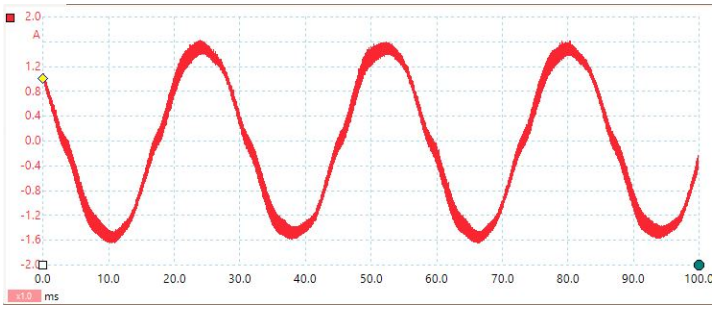


Fig. 4.72 Experimental results: measured phase current, HIL operation

The measured current amplitude is in agreement with the time plot in figure 4.71, which shows the per unit amplitude and is found after division of the actual value by the full-scale current value of 40 A.
The operating speed of the machine can also be deduced from the experimental results, which shows that the electrical period is equal to 28.1 ms, the inverse of which is indeed equal to the applied reference frequency.

Prior to activating this type of example, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter

- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.1.1.11 Voltage & Frequency control-F069-HVkit: HIL

Reference files:

- `V&F_HV_kit_069M-compile.vsm`, for HIL-compile
- `V&F_HV_kit_069M-run_d.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.73 represents the High Voltage Motor Control Kit (linked to the `LAUNCHXL-F28069M` ISO control card), which is connected to a PMsyn machine. For HIL operation the drive board itself needs to be known in considerable detail. Aspects need to be considered such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs, which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.74 shows the 'voltage controller' module, which must be compiled to generate an out-file.

Inputs, shown in this stage as constants (for readability purposes) for this controller module, are the user direct axis reference voltages `ud`,`uq` and the frequency `freq`, which are converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The output of the Controller module is the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts. In addition, a `Test` output (to allow users to add a numeric display or plot to monitor operation) and variable `CPU` is provided, that gives an indication of MCU usage relative to the sampling time.

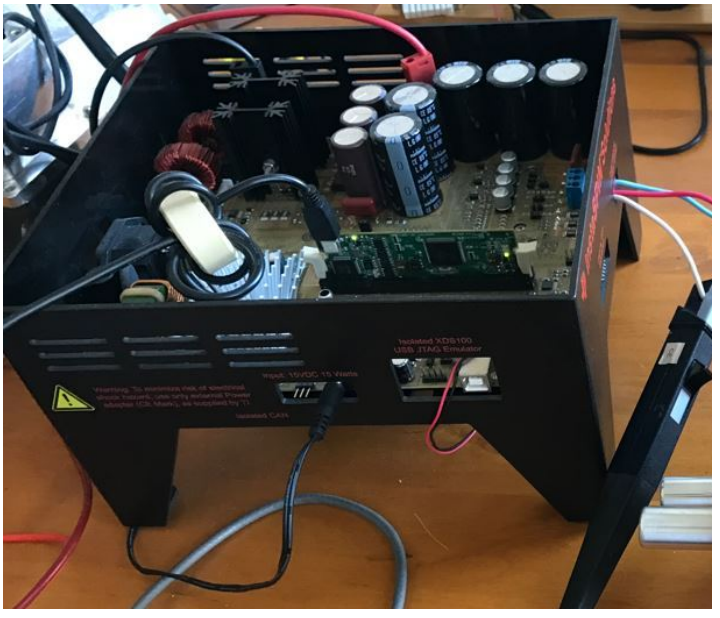


Fig. 4.73 F28069 MCU-HV motor control kit for HIL operation

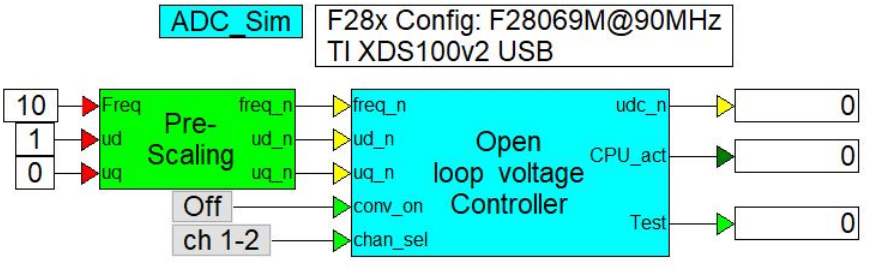


Fig. 4.74 HIL-Compile model of a PMsyn drive with voltage & frequency control

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes, as shown in figure 4.75.
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.76), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage/frequency.
The 'Converter Variables' dialogue box, as given in figure 4.77, defines the scaling used by the Analogue to Digital converter, as discussed earlier (for a

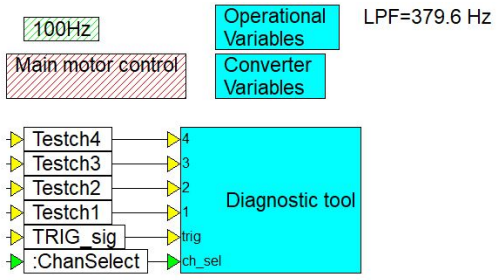


Fig. 4.75 One level down into the 'Open loop Voltage Controller' module

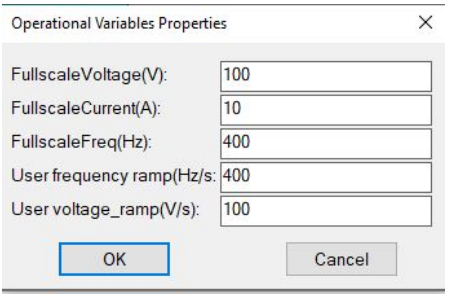


Fig. 4.76 HIL: Dialogue box 'Operational Variables'

different converter), see equation (4.4).

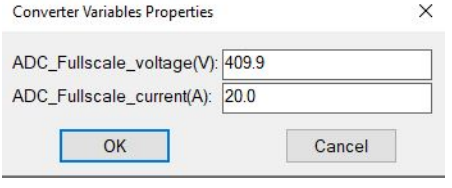


Fig. 4.77 HIL: Dialogue box 'Converter Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component, shown in figure 4.75, is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit.
Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 4.78. Of these, the 'Voltage controller' has been discussed previously, for a different MCU (see figure 4.3), where it was used in sections: SIM, PIL and COSIM of the respective development tool-chain.

The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 10 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 2 μs
- Board in use: `F28069` ISO control card with `HV motor control kit` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

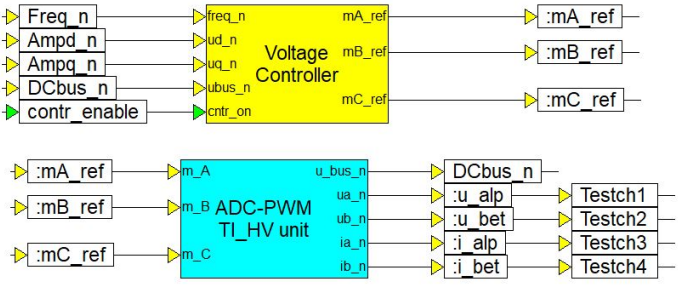


Fig. 4.78 One level down into the 'Main motor controller' module

The `ADC-PWM TI_HV unit` unit is used to generate the PWM and measure the three (per unit) low-pass filtered half-bridge voltages, which are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three (per unit) measured phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively.

ADC Offsets in this unit, are removed with the aid of a low-pass filter,

with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by a '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational phase of this example, shown in figure 4.79 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes said out-file.
Three sliders are used to set the direct/quadrature voltage reference values and the electrical frequency. A post-scaling module is used to convert the per-unit measured DC bus voltage to actual value, as shown with a numeric display. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

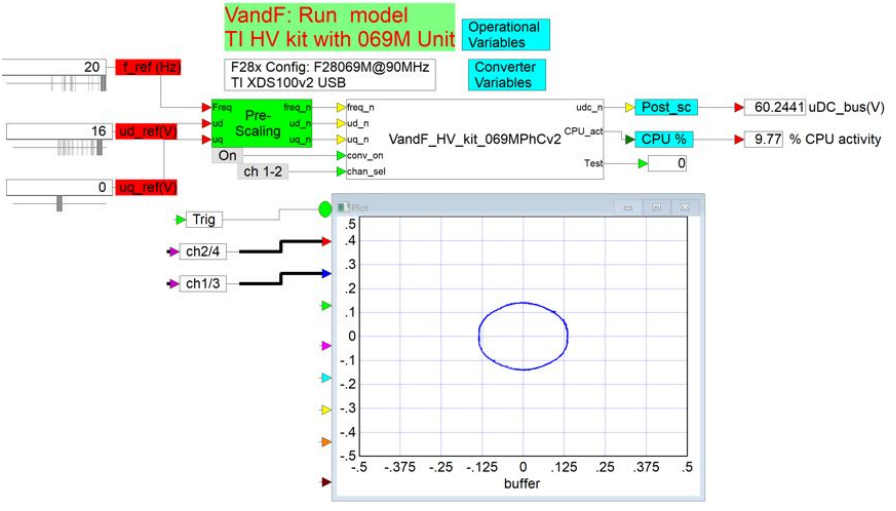


Fig. 4.79 HIL-Debug model, F28069-HV motor control kit of a PMsyn drive with voltage & frequency control

The plot given in figure 4.79 displays the per unit LPF measured voltage vector locus of variables `u_alp`, `u_bet`. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit α, β current vector locus. Note that the voltage reference slider `ud_ref` is set to 16 V (and `uq_ref`=0), hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{16}/_{100} = 0.16$, given that a full-scale voltage of u_{fs}=100 V is used.
Observation of the plot result confirms that the voltage amplitude is approximately equal to the anticipated value. Some amplitude deviation will inevitably occur due to practical issues such as converter dead-time and

voltage drop across the converter switches. The electrical frequency reference in figure 4.49 is set to 20 Hz, which corresponds to a shaft speed equal to $n_m = {}^{60\,f_{ref}}/_{p} = 300$ rpm, given that a four pole pair machine is used, hence $p = 4$. The per unit phase current amplitude is equal to ≈ 0.02, which corresponds to a real value of 0.2 A.

Prior to activating this type of example, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 60 V (for this example). If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.1.1.12 Voltage & Frequency control-F0039-GANINV: HIL

Reference files:

- `V&F_F280039-GANIN-compile.vsm`, for HIL-compile
- `V&F_F280039-GANINV-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.80 represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F280039C` module), which is connected to a PMsyn Teknic motor[2].

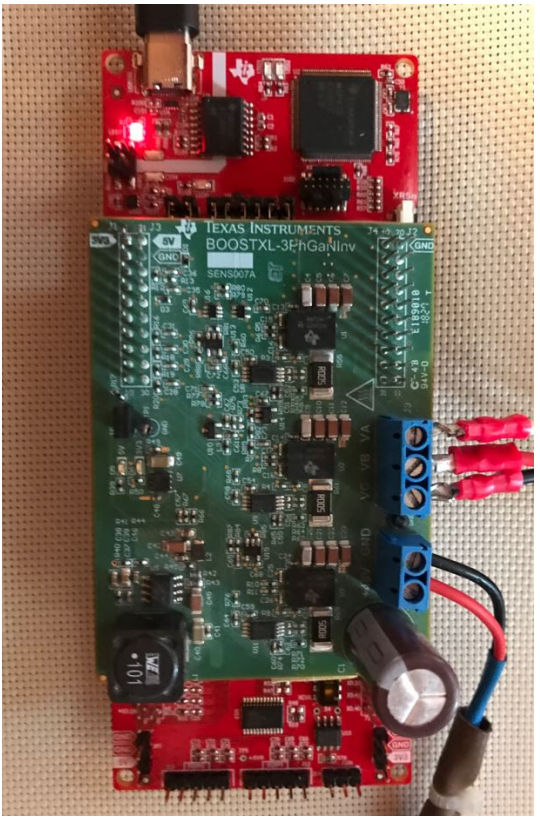


Fig. 4.80 F280039C MCU-GANINV converter setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects need to be considered such as allocation of data acquisition channels relative to the converter voltage and current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate the file `XXX.out`, that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.81 shows the 'open loop voltage controller' module, which must be compiled to generate an out-file.

Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct and quadrature reference voltages `ud`, `uq` and the frequency `Freq`, which are converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed.

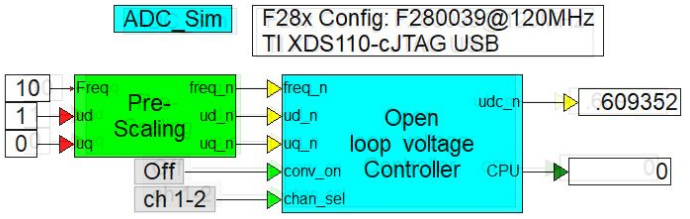


Fig. 4.81 HIL-Compile model of a PMsyn drive with voltage & frequency control

Outputs of the Controller module are the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.

Also shown in figure 4.81 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 4.82.

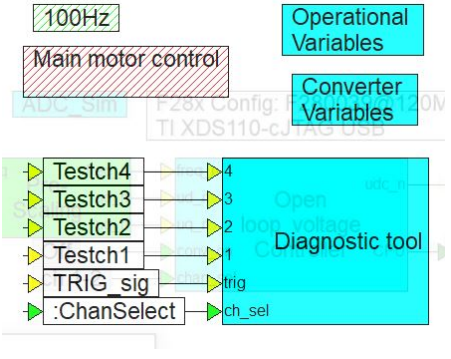


Fig. 4.82 One level down into the 'Open loop Voltage Controller' module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.83), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage/frequency.
The 'Converter Variables' dialog box (see figure 4.84), defines the scaling used

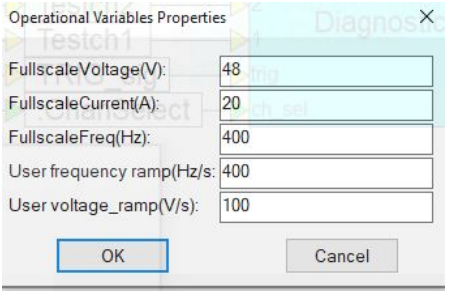


Fig. 4.83 HIL: Dialogue box 'Operational Variables'

by the Analogue to Digital converter, as discussed earlier, see equation (4.4).

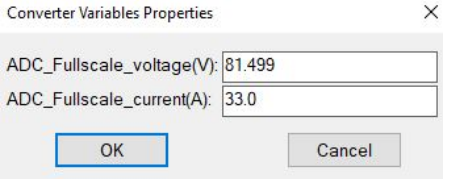


Fig. 4.84 HIL: Dialogue box 'Converter Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.82 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 4.85. Of these, the 'Voltage controller' has been discussed previously, for a different MCU, see figure 4.3, where it was used in phases : SIM, PIL and COSIM of the respective development tool-chain.

The per unit variables: DC bus voltage, half-bridge currents and voltages are

generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F280039` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after every second EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

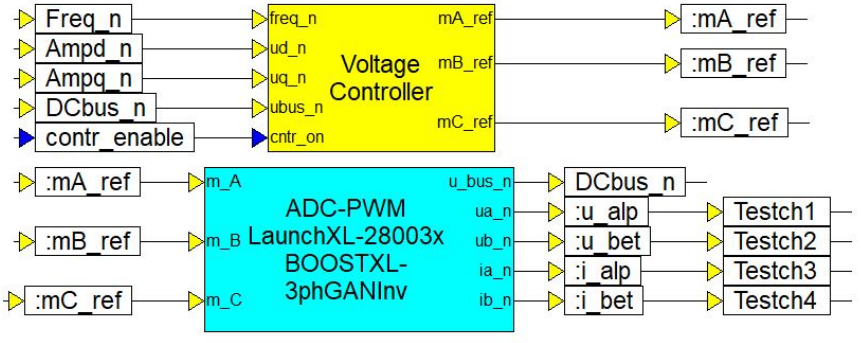


Fig. 4.85 One level down into the 'Main motor controller' module

Moving one level down the `ADC-PWM` unit (given in figure 4.85), is a `PWM unit BOOSTXL-GANINV` module, that controls the converter switches based on the modulation indices generated by the controller (see figure 4.86).
The `ADC BOOSTXL 3PH-GANINV` unit is used to measure the three low-pass filtered half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp_n`, `u_bet_n` using a forward Clarke conversion. Similarly, the three measured phase currents are converted to stationary coordinates α, β with outputs `i_alp_n`, `i_bet_n` respectively.

ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offsets is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `u_DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 4.87 and its task is to execute the `XXX.out` file

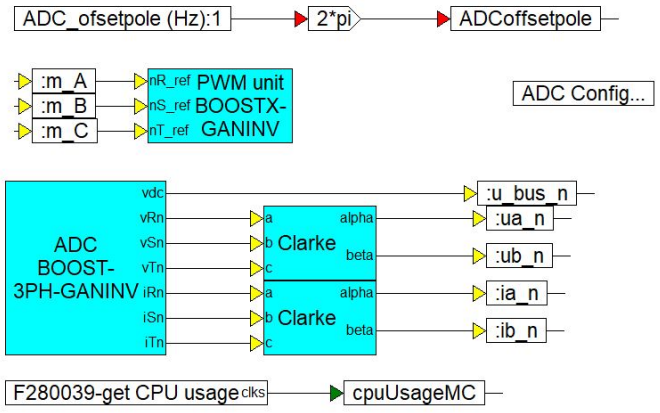


Fig. 4.86 One level down into the 'ADC-PWM' module

compiled in 'HIL-compile' phase (see above).

Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct/quadrature voltage reference values and the electrical frequency. Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays.
Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.
The plot given in figure 4.87, displays the per unit LPF measured voltage vector locus. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus. Note that the voltage reference slider `ud_ref` is set to 1.5 V (and `uq_ref`=0), hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{1.5}/_{48} = 0.031$, given that a full-scale voltage of u_{fs}=48 V is used.
Observation of the plot result confirms that the amplitude is approximately equal to the anticipated value. Some amplitude deviation will inevitably occur due to practical issues such as converter dead-time and voltage drop across the converter switches. The electrical frequency reference in figure 4.87 is set to 32.1 Hz, which corresponds to a shaft speed equal to $n_m = {}^{60\, f_{ref}}/_{p} = 481.5$ rpm, given that a four pole pair machine is used, hence $p = 4$.

For comparative purposes an PicoScope [20] with DC current probe was used to measure the phase 'A' current, shown in figure 4.88. The current amplitude shown in this figure, matches the per unit current locus amplitude (when selecting `ch 3-4` for the scope module), which confirms that the `ADC_fullscale_current` setting is correct.

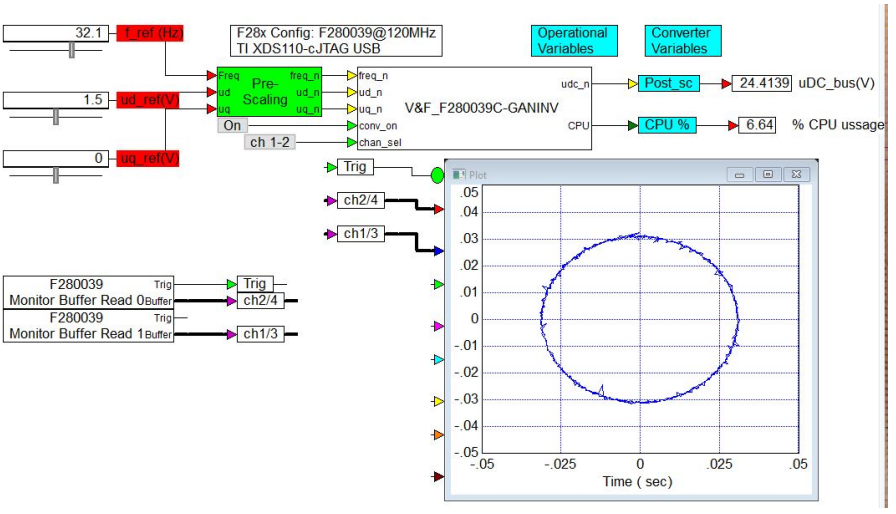


Fig. 4.87 HIL-Debug model, F280039-GANINV module of a PMsyn drive with voltage & frequency control

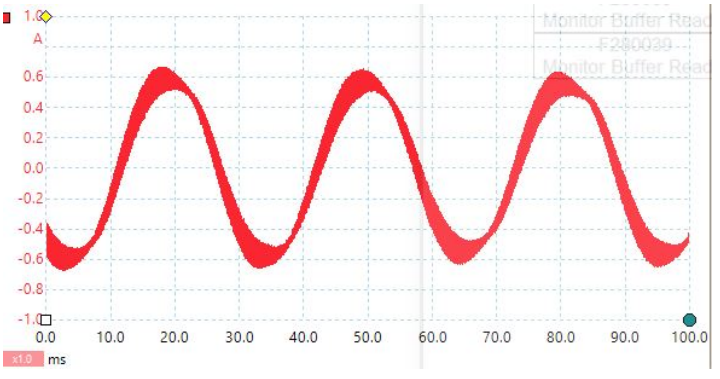


Fig. 4.88 Experimental results: measured phase current, phase 'A'

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.

- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.1.2 Voltage & Frequency Control-STM

The purpose of this section is to realize voltage & Frequency control of a PMsyn machine using ST Micro hardware. At the same time this type of example is used to make a functional check on the current/voltage measurements and overall functioning of the drive board and software. As such this section provides a set of 'foundation' ST Micro examples, which will substantially reduce the development time required. The reason being that users can simply redesign/modify the controller to their needs as the hardware setup phase for HIL's provided is in place. The tool chain as introduced in figure 2.45 is used as a template for the ensuing subsections.

4.1.2.1 Voltage & Frequency control-F303-ESC001: SIM

Reference file: `V&F_F303_ESC001-SIM.vsm`

This example considers the simulation of a PMsyn Teknic motor [2] connected to a ST Micro F303 ESC001 converter, as shown in figure 4.89. The aim is to examine operation of the drive using a fixed point controller, that will subsequently be used in the HIL implementation.

This example makes use of the 'voltage controller' introduced earlier, figure 4.3 (for a different MCU). Outputs of the controller are three modulation indices that control the three converter half bridges.

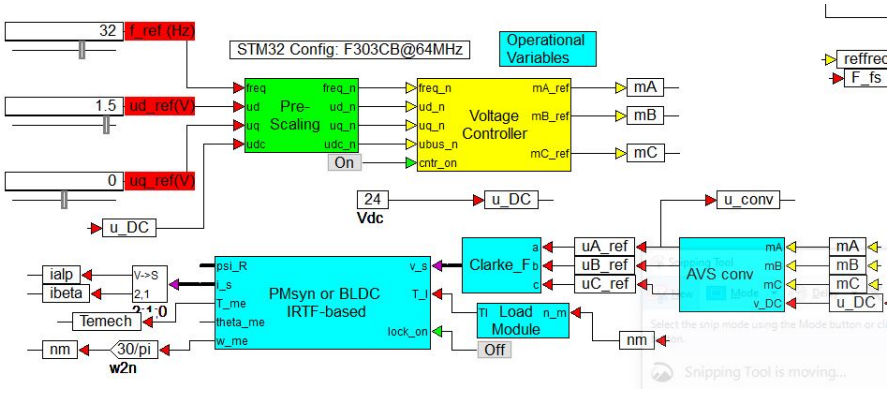


Fig. 4.89 Voltage & Frequency control: simulation

The voltages supplied to the motor `uA_ref`,`uB_ref`,`uC_ref` are found by multiplying the modulation indices `mA`, `mB`, `mC` by half the DC bus voltage, as executed in module `AVS conv`, see figure 4.89. These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables. The aim of the switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sub chapter 2.1.5). A quadratic load is introduced, which provides a value of 10mNm at a shaft speed of 1000rpm.

The IRTF based PM model [13] has a user option to select a synchronous or BLDC type machine, as discussed earlier (see figure 2.12). When selecting the 'BLDC' option a six harmonic component is added to the PM flux locus.

The results shown in figure 4.90, confirm the presence of voltage and frequency control. Note that at start up, ramp functions are used, which determine the rate of reference voltage and frequency variable changes, as set in the 'Operational Variables' dialogue box.

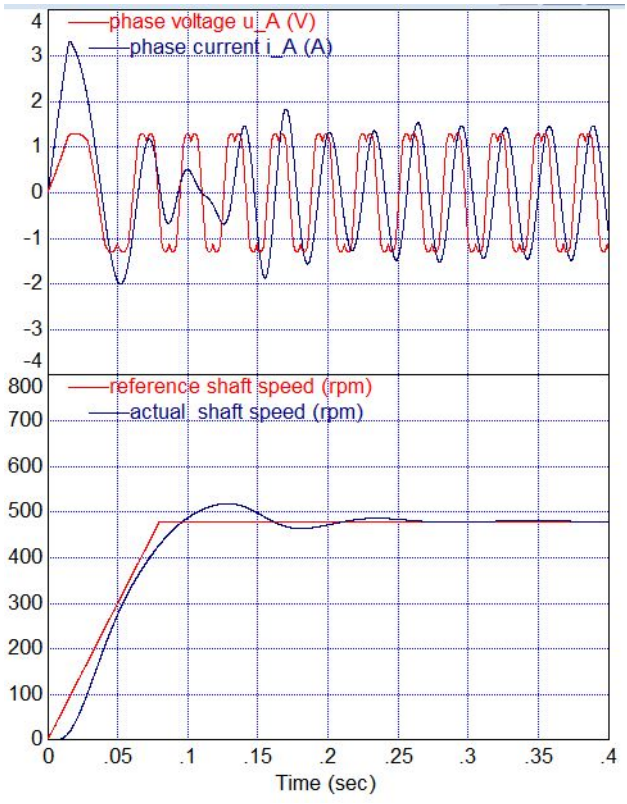


Fig. 4.90 SIM result, PMsyn operation, voltage & frequency control

4.1.2.2 Voltage & Frequency control-F303-ESC001: PIL

Reference files:

- `V&F_F303_ESC001-PIL.vsm`
- `V&F_PIL_mod-ST.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F303. This implies that the controller module shown in the simulation (see figure 4.89) is now replaced with a file: `V&F_PIL_mod_ST.out`, as indicated in figure 4.91.

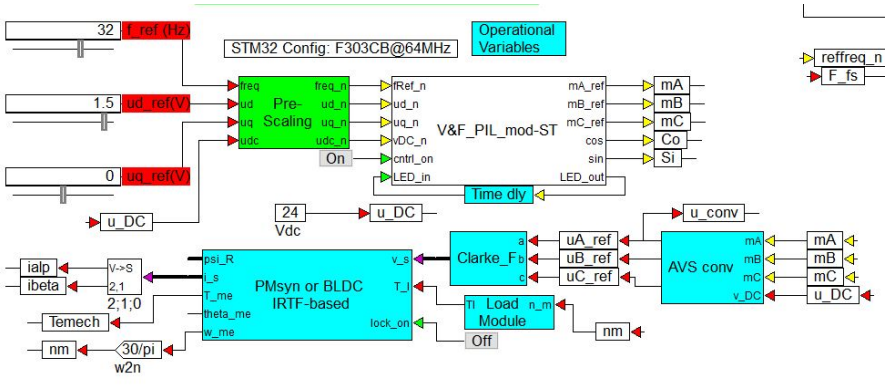


Fig. 4.91 PIL based drive, PM syn operation, voltage & frequency control

The target interface model that executes the out-file, generates the modulation indices required for the converter motor and these components are not affected by use of the PIL approach. Generation of the out file is based on the PIL module (see figure 4.92), which must be compiled using the command: `toolbox>codegen`.

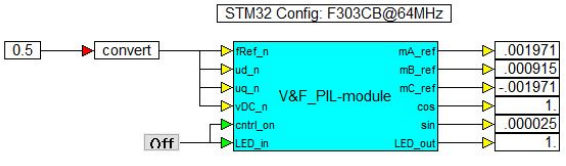


Fig. 4.92 STM PIL module, voltage & frequency control

Moving one level into the PIL module `V&F_PIL_mod` shows the voltage controller, as introduced in section 4.1.2.

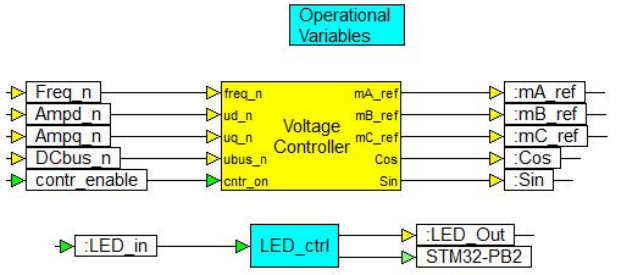


Fig. 4.93 Content STM PIL module, voltage & frequency control

Added to the voltage controller is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that the PIL control is operational. In addition, two variables `Cos`, `Sim` are provided as outputs and these are used in a vector plot to show the 'd' axis of the controller algorithm.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the ST board, which indicates that the MCU is operational. Results produced by PIL operation are identical to the plot shown in figure 4.90 and is therefore not shown here.

4.1.2.3 Voltage & Frequency control-F303-ESC001:COSIM

Reference files:

- `V&F_F303_ESC001-COSIM.vsm`
- `ESC001_TeknicPM_mot.psimsch`
- `my motor model_v3.psimsch`

To evaluate operation of a controller with a switching converter, in this case the `ESC001`, use can be made of a `PSIMCoupler` module, as indicated in figure 4.94. This approach implies the use of PSIM together with Embed, which is referred to as 'co-simulation'.
The `Voltage Controller` module, hatched in this case, as its sampling frequency is locally set to 10 kHz, whereas the 'system' frequency (set in the 'system properties' dialogue box) is set to20 Mhz. The latter is much higher, given the need to accurately represent 30 kHz voltage switching waveforms generated by the PWM unit of the converter. The content of the `Voltage controller` module was introduced earlier, see figure 4.3 (for a different MCU). Furthermore, the current and voltage measurements now make use of the signal processing circuitry connected to the half-bridges.

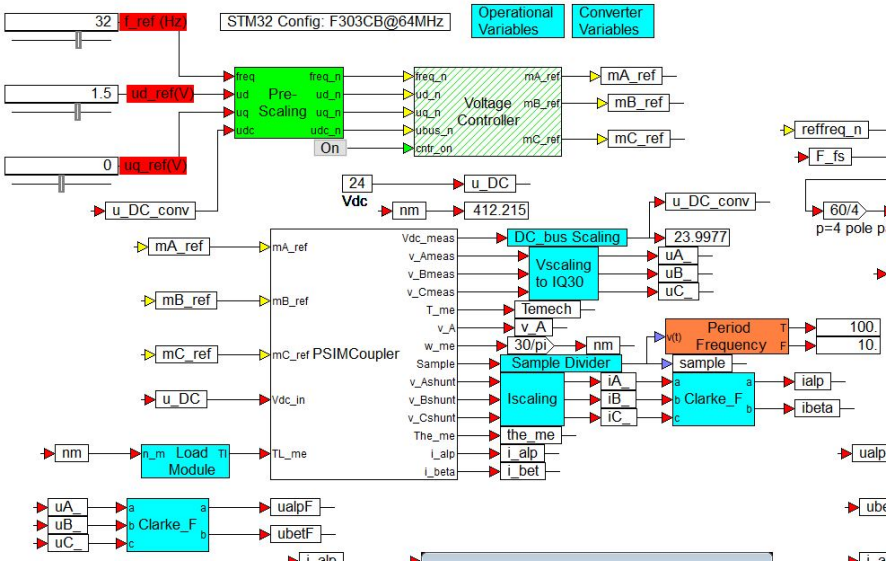


Fig. 4.94 COSIM based drive, voltage & frequency control

In this case the current shunts are connected to the lower converter switches, as shown in figure 4.95. After sampling and signal-processing in the converter, these variables appear as: `iA_`,`iB_`,`iC_` in figure 4.94. The low pass filtered voltages (which removes the PWM component) of the three half-bridges is also present in figure 4.94 and are also scaled and sampled by the module `Vscaling to IQ30`. In this case the per unit variables are rescaled to actual values: `uA_`, `uB_` and `uC_`.
Further observation of figure 4.95 shows the presence of the module `PMSM`, which is a generic IRTF model of the IPM synchronous machine machine, as presented earlier (see figure 2.17). However, the PSIM model, introduced earlier (see figure 4.11), has an 'electrical interface' to allow use of said model with the converter.

A PWM unit referred to as `3-ph PWM`, generates the gate control signals used to switch on/off the upper or lower devices of each converter leg. Connected to the output of each converter terminal is a resistance/capacitor network that is used to measure and low-pass filter the converter output voltage in order to eliminate the PWM switching effects, as mentioned earlier.

Central to sensorless operation is the ability to accurately measure the phase voltages and currents. The experimental drive board(s) used in this GUIDE are mostly 'sensorless prepared', which implies that they have been purposely designed with a data acquisition module (ADC) that undertakes the sig-

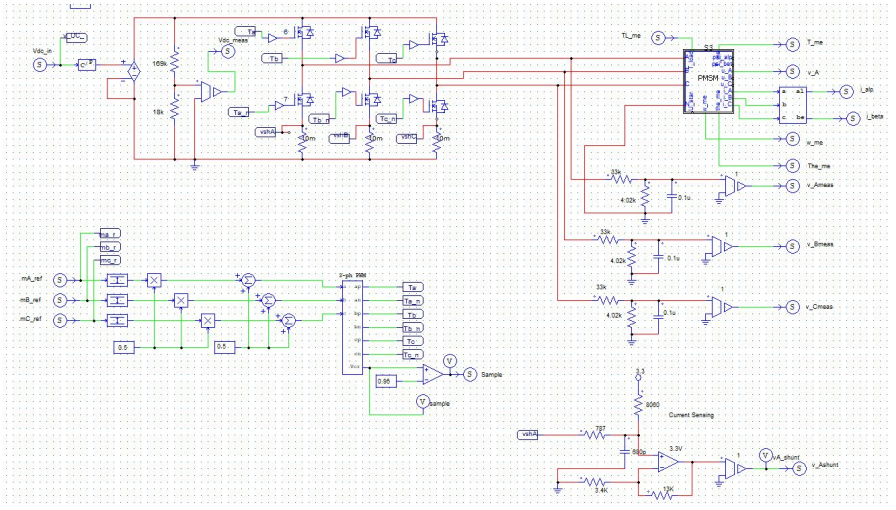


Fig. 4.95 Content of the PSIM coupler module, voltage & frequency control

nal conditioning of the voltage/currents as needed to realize sensorless (no mechanical encoder attached) control. In addition, access to voltages and currents, gives the user ability to estimate the motor input power, which in turn provides an indication of the DC bus current. The latter therefore, provides an estimate for the energy usage e.g. *Ah* of battery operated drives. The converter model discussed here is 'sensorless prepared' and at this stage it is helpful to examine the data signal condition of the circuit in more detail, where use is made of figure 4.96. This is required because a number of critical hardware based parameters must be provided to the controller, by the user to ensure correct operation of the drive.

Readily apparent in figure 4.96 is the resistor/capacitor network used to attenuate and low-pass filter the converter output voltage to be used by the controller. The attenuation/gain G_R and corner frequency ω_c of the low pass filter circuit are equal to:

$$G_R = \frac{R_4}{R_3 + R_4} \tag{4.8a}$$

$$\omega_c = \frac{R_3 + R_4}{R_3 R_4 C_4} \tag{4.8b}$$

The corner frequency ω_c (rad/s) and attenuation of this filter are board dependent, hence their value must be identified by inspection of the components (or provided via the circuit diagram).

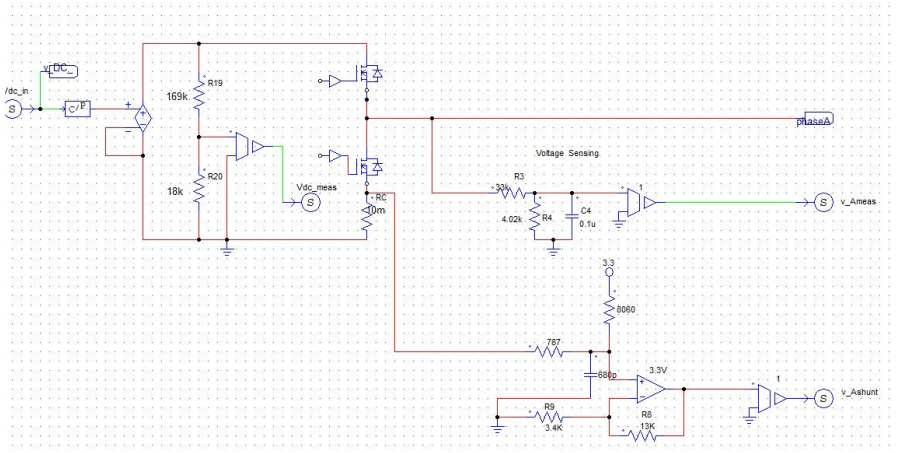


Fig. 4.96 One (of three) phase PSIM model of the ESC001 unit

The example given in figure 4.96 shows the voltage filter components as present in the `ESC001` module, which provide an attenuation of $G_R = 0.108$ and $\omega_c = 2790$ rad/s respectively. Note that the DC bus voltage measurement for the `ESC001` has a different attenuation, when compared against those used for the converter LP filters, hence the ADC fullscale voltage for the DC bus will also be different. The low-pass filtered converter voltage `v_Ameas`, `v_Bmeas` and `v_Cmeas`, are sampled by the `Vscaling to IQ30`module shown in figure 4.9. When dealing with the actual MCU, the ADC converter takes over this sampling process.

An important part of the data acquisition process is measurement of the phase current with the aid of a shunt resistor `RC` (see figure 4.96). The voltage across this shunt is amplified by an op-amp circuit with gain G_{sh}, hence the voltage output `v_Ashunt` for the MCU (and an output of the PSIM module) is equal to:

$$v_{\text{shunt}} = i_{\text{conv}}\, R_{\text{RC}}\, G_{\text{sh}} \tag{4.9}$$

where, R_{RC} represents the resistance 'RC', shown in figure 4.96.
The 'Converter Variables' dialogue box (see figure 4.9), defines the scaling used by the Analog to Digital converter, which has a maximum input value of 3.3 V and is directly linked to the signal processing undertaken for voltages and currents as discussed above. For example, the 'ADC-fullscale-voltage' value is the theoretical peak-to peak input voltage which must be applied to

the low-pass filter network (see figure 4.96) to ensure that the ADC input level varies between $0 \rightarrow 3.3$V. Subsequent use of equations (4.2),(3.3) shows that these values are defined as:

$$\text{ADC} - \text{fullscalevoltage} = \frac{3.3}{G_{\text{R}}} \tag{4.10a}$$

$$\text{ADC} - \text{fullscalecurrent} = \frac{3.3}{R_{\text{RC}}\, G_{\text{sh}}} \tag{4.10b}$$

where G_{R}, is the (zero frequency) low-pass filter attenuation used for measuring the converter voltages. Similarly, the 'ADC-fullscale-current' value is the peak to peak current in the shunt resistance, which corresponds to an ADC input voltage range of $0 \rightarrow 3.3\,\text{V}$. Hence its value is determined by the shunt resistance value R_{RC} and the signal amplification gain G_{sh}, between said resistor and ADC input. The voltage and current measurements are unipolar and bipolar respectively.
This implies that the voltage measurement is referenced to the negative converter DC bus voltage, whilst the current is referenced to the center of the ADC voltage span, i.e $^{3.3}/_{2}\,\text{V}$, as provided by an op-amp.
Note that the ADC dialogue values, are board dependent and need to be found by inspection of the circuit (or provided). The current shunt voltage(s) `v_Ashunt` are sampled by the `Iscaling` module shown in figure 4.94. When dealing with the actual MCU, the ADC converter also takes over this sampling process. Apart from the measured voltage/currents generated by the converter, the PSIM coupler unit also provides the actual motor model current vector:`i_alp`, `i_bet` (in stationary coordinates) as well as the phase 'A' voltage and speed, which are presented by the variables `v_A` and `n_m` (after conversion to rpm) respectively (see figure 4.94).

Results achieved with the COSIM approach are shown in figure 4.97.
Indicated in said figure is the phase 'A' voltage waveform together with the reference voltage, phase current, shaft torque and shaft speed. A vector plot (see figure 4.98), at time mark 68 ms after starting the co-simulation shows measured LPF voltage vector ('blue'), unity d-axis vector ('green') and the actual (from the IRTF motor model) current vector ('red'). Note that the impact of switching is clearly reflected in the current vector locus, e.g. changes in end-point direction due to use of 'active' or 'zero' voltage vectors [13].
Co-simulation, as discussed in this sub-section, is instructive for understanding the impact of using a switching converter on drive operation.

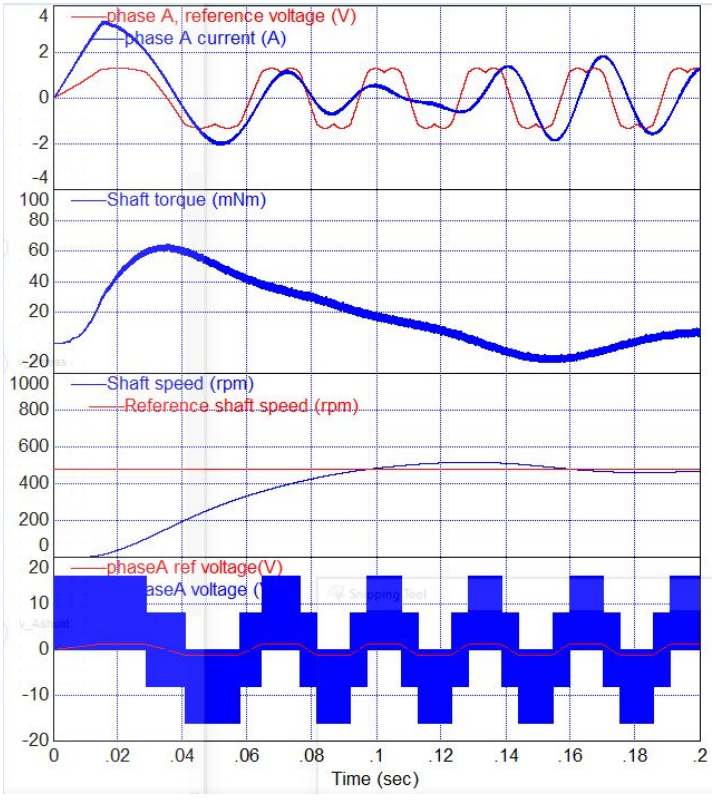


Fig. 4.97 Result using COSIM, voltage & frequency control PMsyn drive

4.1.2.4 Voltage & Frequency control-F303-ESC001: HIL

Reference files:

- `V&F_F303RE_ESC001_compile.vsm`, for HIL-compile
- `V&F_F303RE_ESC001_run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.99 represents the `ESC001` module, which is connected to a PMsyn Teknic motor[2].
For HIL operation, the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration.

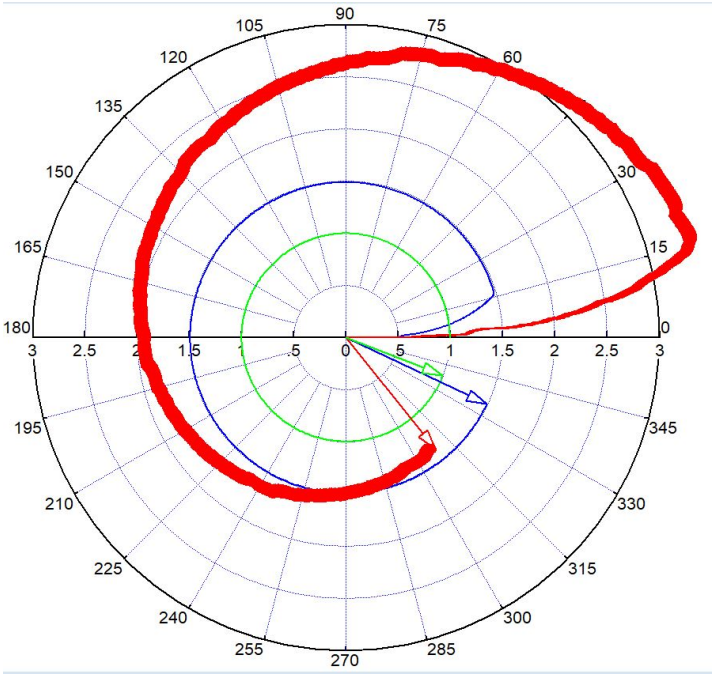


Fig. 4.98 Result using COSIM, voltage & frequency control PMsyn drive: vector plot

Fig. 4.99 STM F303-ESC001 setup for HIL operation

The model given in figure 4.100 shows the 'voltage controller' module, which must be compiled to generate an out-file.

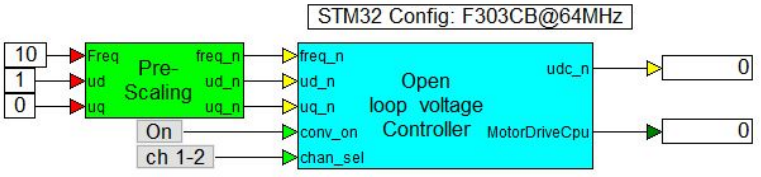


Fig. 4.100 HIL-Compile model of a PMsyn drive with voltage & frequency control

Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct and quadrature reference voltages `ud`, `uq` and the frequency `Freq`, which are converted to 'per unit' values in the 'pre-scaling' module.
Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Output of the controller module is the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts. Furthermore, a `CPU` output is added, which provides the user with a % reading of MCU activity, relative the controller sample time.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 4.101.

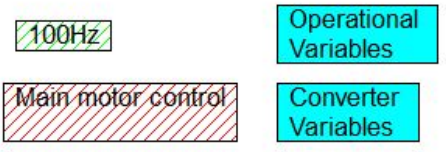


Fig. 4.101 One level down into the 'Open loop Voltage Controller' module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.102), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage-frequency.

The 'Converter Variables' dialogue box (see figure 4.103), defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equa-

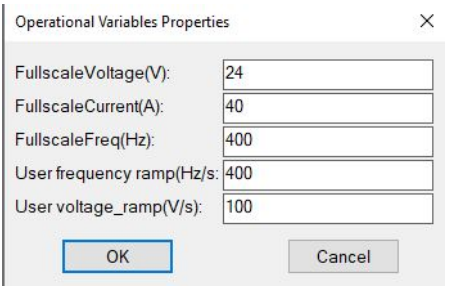


Fig. 4.102 HIL: Dialogue box 'Operational Variables'

tion (4.10).

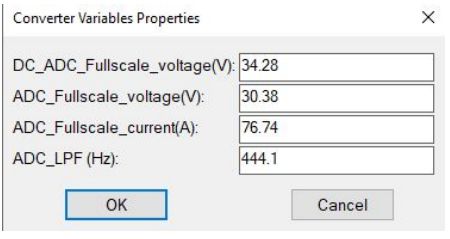


Fig. 4.103 HIL: Dialogue box 'Converter Variables'

Note that this dialog box shows two separate ADC fullscale voltage values, one for the converter and one for the DCbus, as was shown earlier(see figure 4.96). A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in HIL-debug mode.
A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.101 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 4.104. Of these, the 'Voltage controller' has been discussed previously,

for a different MCU (see figure 4.3), where it was used in tool-chain elements: SIM, PIL and COSIM of the respective development tool-chain.

The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 30 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0.2 μs
- Board in use: `ESC001` converter with `F303` MCU

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after every third (for this drive) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

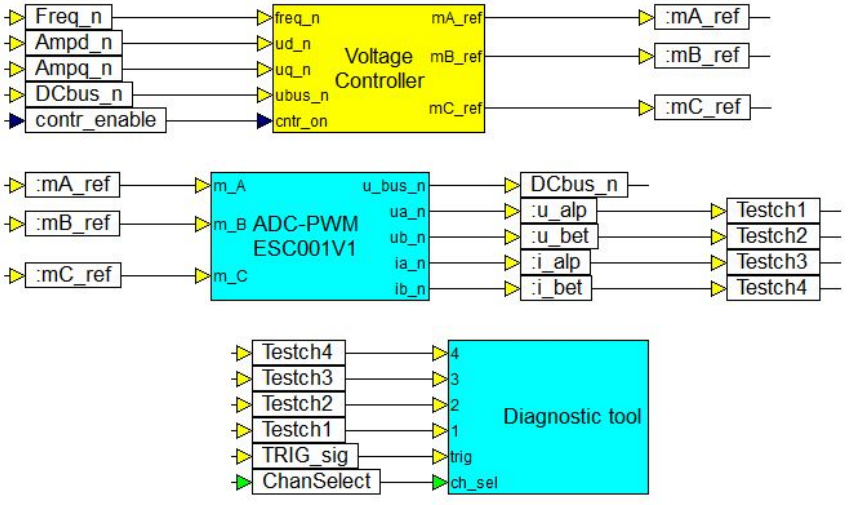


Fig. 4.104 One level down into the 'Main motor controller' module

The `ADC-PWM ESC001` unit is used to generate the PWM and to measure the three low-pass filtered half-bridge voltages, which are converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured phase currents are converted to stationary coordinates α, β, with outputs `i_alp`, `i_bet` respectively.

ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box

of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 4.105 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).

Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct and quadrature voltage reference values and the electrical frequency. Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

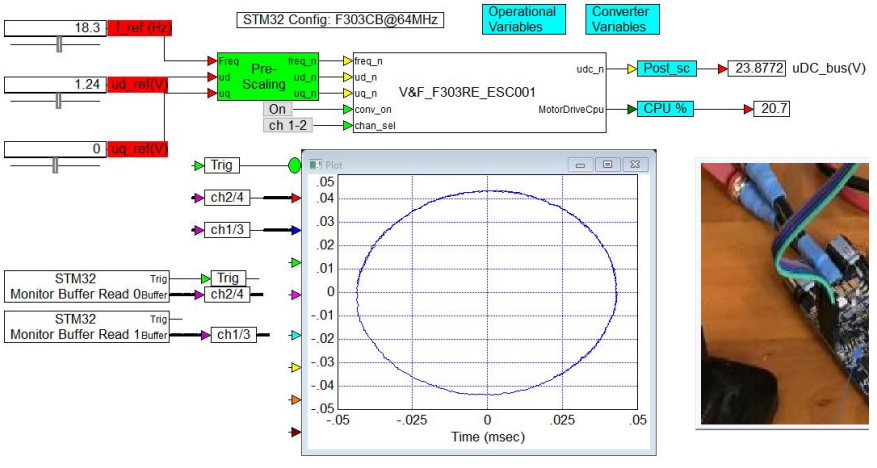


Fig. 4.105 HIL-Debug model, F303-ESC001 of a PMsyn drive with voltage & frequency control

The plot given in figure 4.105 displays the per unit LPF measured voltage vector locus. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus. Note that the voltage reference slider `ud_ref` is set to 1.24 V (and `uq_ref`=0), hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{1.24}/_{24} = 0.051$, given that a full-scale voltage of u_{fs}=24 V is used. Observation of the plot result confirms that the amplitude is approximately equal to the anticipated value. Some amplitude deviation will inevitably occur due to practical issues such as converter dead-time and voltage drop across the converter switches. The electrical frequency reference in figure 4.105 is set to 18.3 Hz, which corresponds to a

shaft speed equal to $n_m = {}^{60\,f_{ref}}/_{p} = 274$ rpm, given that a four pole pair machine is used, hence $p = 4$.

For comparative purposes an oscilloscope [20] with DC current probe was used to measure the phase 'A' current, as shown in figure 4.106.

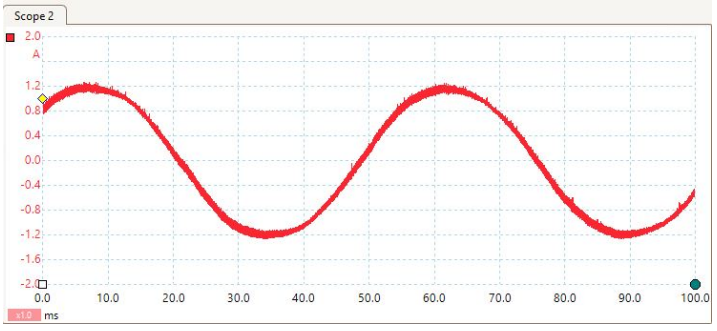


Fig. 4.106 experimental results: measured phase current, phase 'A'

The measured current amplitude is 1.1 A, which corresponds to a per unit value of ${}^{1.25}/_{40} = 0.031$ which is is agreement with the current locus in figure 4.105 (shown on the plot when selecting channel `3-4`).

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm

that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic plot.

4.1.2.5 Voltage & Frequency control-F405-VESC6-MKIII: HIL

Reference files:

- `V&F_VESC6mrk3_compile.vsm`, for HIL-compile
- `V&F_VESC6mrk3_run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.107 represents the `VESC6-MKIII` module which is connected to a PMsyn Teknic motor[2]. This is a highly compact unit that can be used in the voltage range 11.1-60 V and has a 80 A current rating [26].

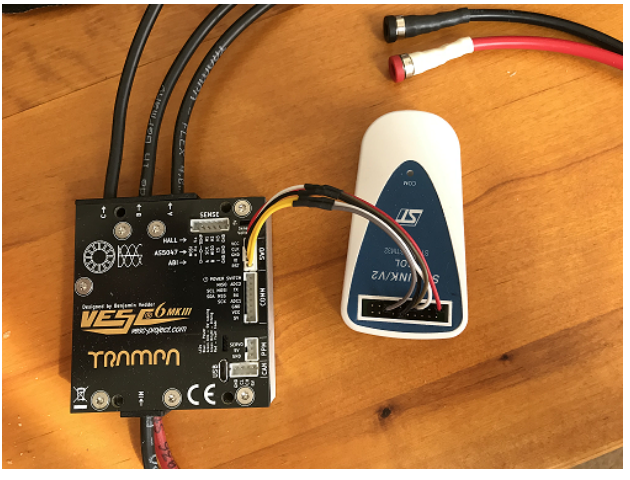


Fig. 4.107 F405 MCU-VESC6-MKIII converter setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.108 shows the 'voltage controller' module, which must be compiled to generate an out-file.

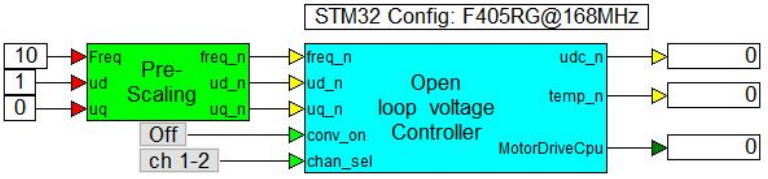


Fig. 4.108 HIL-Compile model of a PMsyn drive with voltage & frequency control

Inputs, shown in the stage as constants (for readability purposes), for this controller module are the user direct and quadrature reference voltages `ud`, `uq` and the frequency `Freq`, which are converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Further information on this tool will be given at a later stage. Output of the Controller module is the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts. Furthermore, a `CPU` output and temperature output `temp_n` is added, which provides the user with a % reading of MCU activity (relative the controller sample time) and module temperature respectively.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes shown in figure 4.109.

Fig. 4.109 One level down into the 'Open loop Voltage Controller' module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.110), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage-frequency.

The 'Converter Variables' dialogue box, as given in figure 4.111, defines the scaling used by the Analogue to Digital converter, for this VESC module. Also added to the dialogue box is the corner frequency of the low-pass filters used to measure the converter voltages.
A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in the HIL-debug mode. A 'two

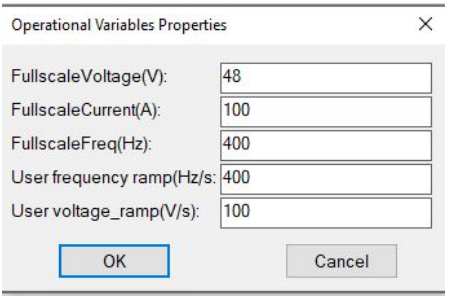


Fig. 4.110 HIL: Dialogue box 'Operational Variables'

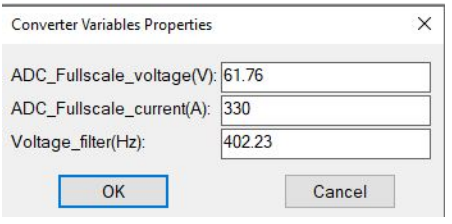


Fig. 4.111 HIL: Dialogue box 'Converter Variables'

channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4`, by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes when the MCU is active.

The key component shown in figure 4.109 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 4.112). Of these, the 'Voltage controller' has been discussed previously, for a different MCU (see figure 4.3), where it was used in sections: SIM, PIL and COSIM of the respective development tool-chain.
The per unit variables, DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 10 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs

- Board in use: `VESC6 MKIII` with `F405` MCU converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

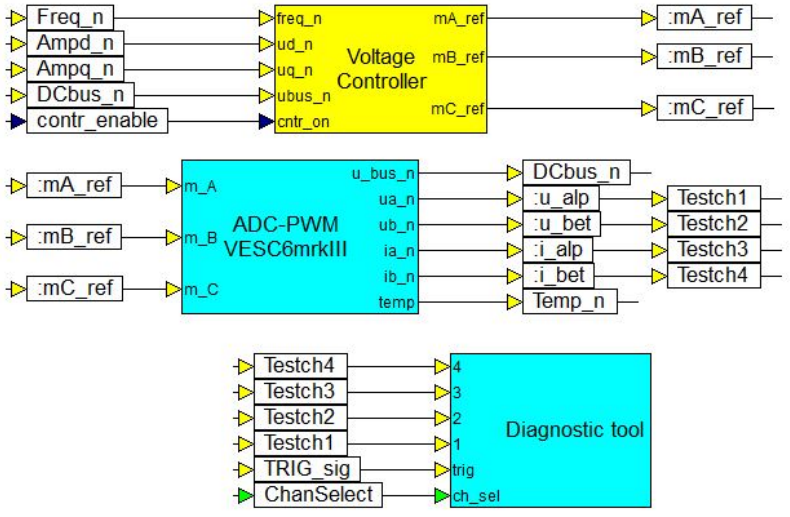


Fig. 4.112 One level down into the 'Main motor control' module

The `ADC-PWM VESC6mrkIII` unit is used to generate the PWM and to measure the three low-pass filtered half-bridge voltages which are converted to stationary coordinate (α, β) voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured phase currents are converted to stationary coordinate (α, β) with outputs `i_alp`, `i_bet` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n` and temperature sensing `temp`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 4.113 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. Three sliders are used to set the direct/quadrature voltage reference values and the electrical frequency. Post-scaling modules are used

to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

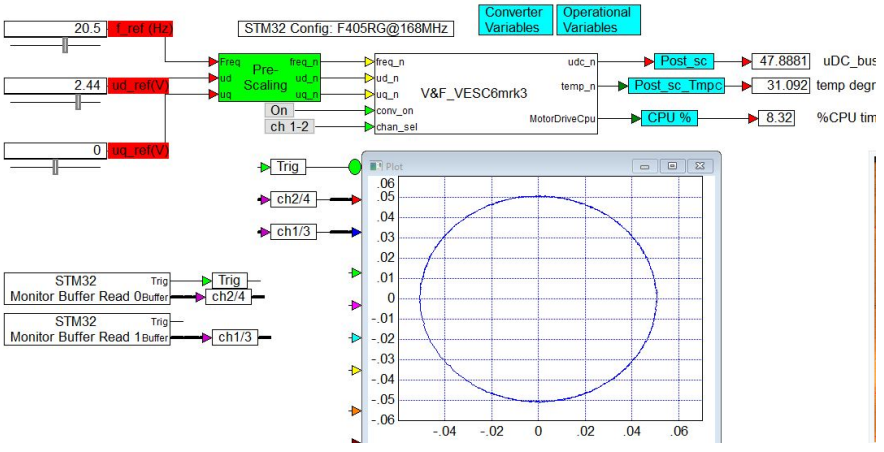


Fig. 4.113 HIL-Debug model, F405-VESC6 MKIII of a PMsyn drive with voltage & frequency control

The plot given in figure 4.113 displays the per unit LPF measured voltage vector locus. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus. Note that the voltage reference slider `ud_ref` is set to 2.44 V (and `uq_ref`=0), hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{2.44}/_{48} = 0.05$, given that a full-scale voltage of u_{fs}=48 V is used. Observation of the plot result confirms that the amplitude is approximately equal to the anticipated value. Some amplitude deviation will inevitably occur due to practical issues such as converter dead-time and voltage drop across the converter switches. The electrical frequency reference in figure 4.113 is set to 20.5 Hz, which corresponds to a shaft speed equal to $n_m = {}^{60\, f_{ref}}/_{p} = 307.5$ rpm, given that a four pole pair machine is used, hence $p = 4$.

For comparative purposes an oscilloscope [20] with DC current probe was used to measure the phase 'A' current, as shown in figure 4.114.
The measured current amplitude is 4.6 A, which corresponds to a per unit value of ${}^{4.6}/_{100} = 0.46$ which is in agreement with the current locus in figure 4.113 (shown on the plot when selecting channel `3-4`).

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with

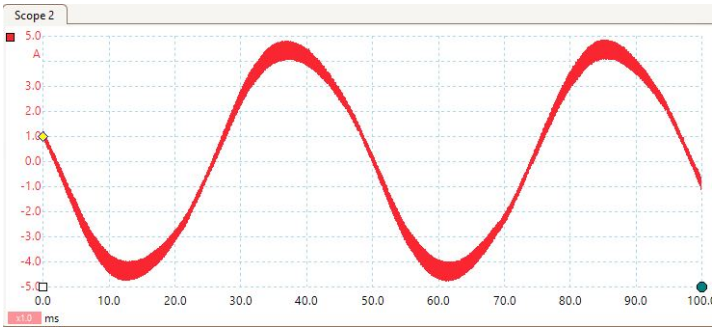


Fig. 4.114 Experimental results: measured phase current, phase 'A'

live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 48 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.1.2.6 Voltage & Frequency control-G473E-B-DRIVE: HIL

Reference files:

- `VandF_G4_Bdrive-compilev4.vsm`, for HIL-compile
- `VandF_G4_Bdrive-debugv4.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.115 represents the `B-G473E-ZEST` board, with converter [34][35], which is connected to a PMsyn Teknic motor[2]. This utilizes a high performance MCU board, with a converter that can be used in the voltage range 6-45 V and has a 4 A current rating. The B-drive ('green board') makes use of current shunts in the lower switches and the voltages across these shunts is amplified using OP-AMP's, before being fed to the ADC inputs. The option for this drive is to either software configure the OP-AMP setup, or hard-wire these. For this example, the hard-wired option has been selected, hence care should be take to configure this board correctly, prior to use.

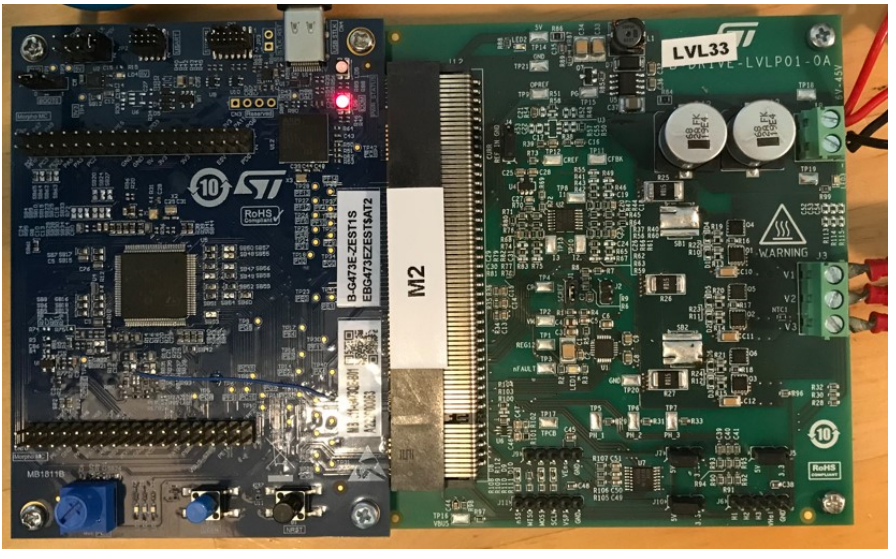


Fig. 4.115 STM B-G473E-ZEST1S and B-DRIVE-LVLP01, setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.116 shows the 'voltage controller' module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes), for this controller module are the user direct and quadrature reference voltages `ud`, `uq` and the frequency `Freq`, which are converted to 'per unit' values in the

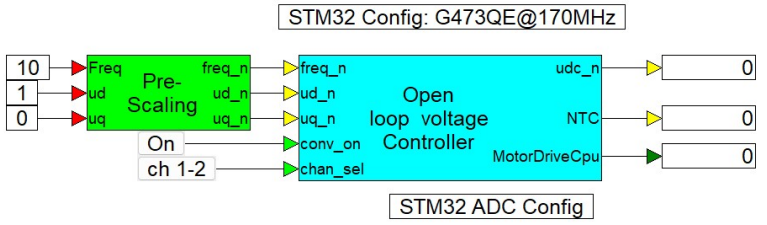


Fig. 4.116 HIL-Compile model of a PMsyn drive with voltage & frequency control

'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Further information on this tool will be given at a later stage. Outputs of the controller module are the per unit DC bus voltage `udc_n`, and drive module temperature `NTC`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts and converter temperature respectively. Furthermore, a `MotorDriveCpu` output is added, which provides the user with a % reading of MCU activity (relative the controller sample time).

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes shown in figure 4.117.

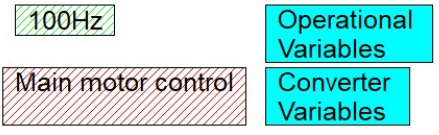


Fig. 4.117 One level down into the 'Open loop Voltage Controller' module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.118), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage-frequency.

The 'Converter Variables' dialogue box (see figure 4.119), defines the scaling used by the Analogue to Digital converter, for this B-drive. Also added to the dialogue box is the corner frequency of the low-pass filters used to measure the converter voltages.
A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in the HIL-debug mode. A 'two

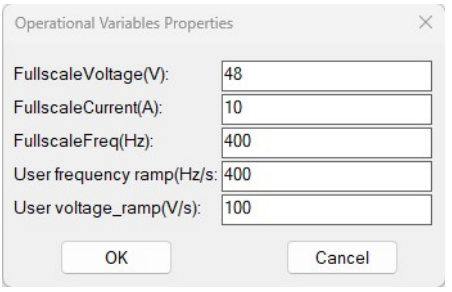


Fig. 4.118 HIL: Dialogue box 'Operational Variables'

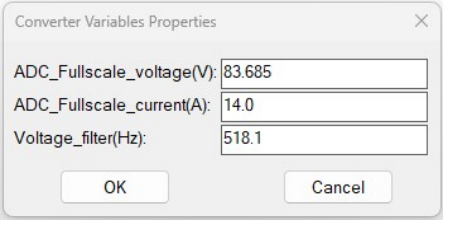


Fig. 4.119 HIL: Dialogue box 'Converter Variables'

channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4`, by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes when the MCU is active.

The key component shown in figure 4.117 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 4.120. Of these, the 'Voltage Controller' has been discussed previously, for a different MCU (see figure 4.3), where it was used in sections: SIM, PIL and COSIM of the respective development tool-chain.
The per unit variables, DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0.2 μs

- Board in use: `B-DRIVE` with `G473QE` MCU converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after every second (for this example) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

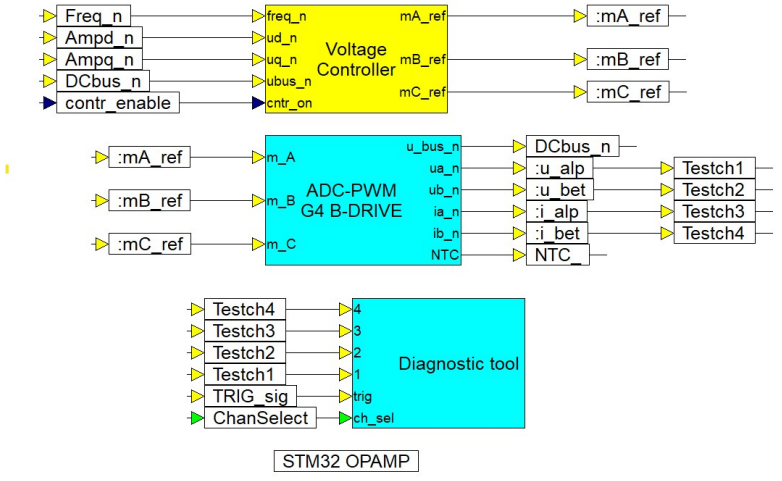


Fig. 4.120 One level down into the 'Main motor control' module

The `ADC-PWM G4 B-DRIVE` unit is used to generate the PWM and to measure the three low-pass filtered half-bridge voltages which are converted to stationary coordinate (α, β) voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured phase currents are converted to stationary coordinates (α, β) with outputs `i_alp`, `i_bet` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n` and temperature sensing `NTC_`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 4.121 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. Three sliders are used to set the direct/quadrature voltage reference values and the electrical frequency. Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Two 'Monitor Buffer' modules are

used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

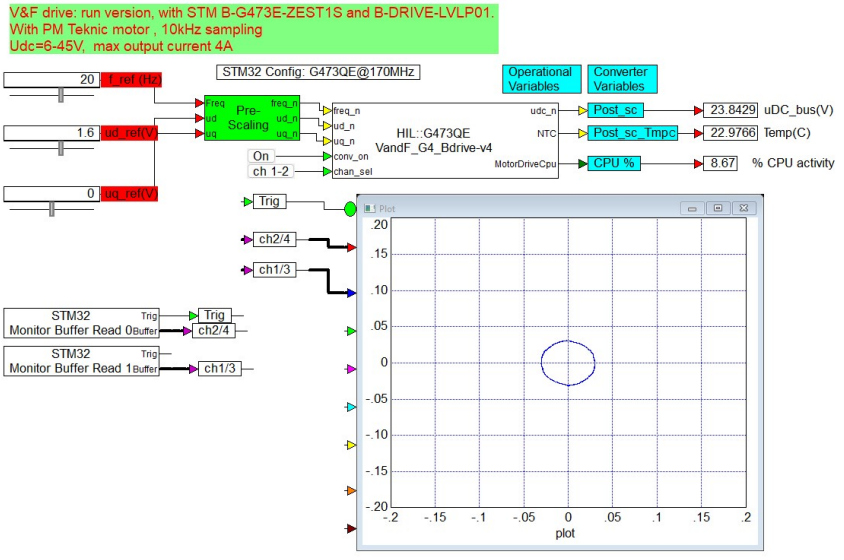


Fig. 4.121 HIL-Debug model, STM B-G473E-ZEST1S and B-DRIVE-LVLP01 of a PMsyn drive with voltage & frequency control

The plot given in figure 4.121 displays the per unit LPF measured voltage vector locus. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus. Note that the voltage reference slider `ud_ref` is set to 1.6 V (and `uq_ref`=0), hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{1.6}/_{48} = 0.033$, given that a full-scale voltage of u_{fs}=48 V is used. Observation of the plot result confirms that the amplitude is approximately equal to the anticipated value. Some amplitude deviation will inevitably occur due to practical issues such as converter dead-time and voltage drop across the converter switches. The electrical frequency reference in figure 4.121 is set to 20.0 Hz, which corresponds to a shaft speed equal to $n_m = {}^{60\,f_{ref}}/_{p} = 300$ rpm, given that a four pole pair machine is used, hence $p = 4$.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.2 Current & Frequency Control

Purpose of this section is to realize open-loop current & frequency control of a PM motor. This may be achieved by using a model based, synchronous frame current controller, as discussed in subsection 2.1.4. Note that the term 'open-loop' refers to operation where the PM machine operates (under closed-loop current control) without any form of rotor position feedback, as was case for the previous section. Like the previous case, the aim is to produce a rotating 'd,q' coordinate system, of which the rotational speed is set by the user reference frequency f^{ref}. The integral of electrical speed forms the rotational angle θ shown in figure 4.122.
Said variable θ, is also the angle between the stationary 'α, β' and 'd,q' coordinate systems. The orientation of the current vector relative to the synchronous coordinate system is determined by the user defined direct and quadrature reference values. In this case the current vector $\vec{i}^{\text{ref}}$ is conveniently aligned with the 'd-axis', as may be observed from figure 4.122. Note that for machine operating under no-load the PM rotor flux IS aligned with the current (no torque produced), which is an important fact in terms of understanding drive operation. When the mechanical load torque is increased the flux vector will lag behind the current vector until the so called 'pull-out torque' is

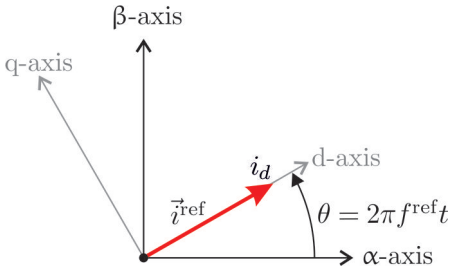


Fig. 4.122 Reference current vector plot

reached, when the angle between said vectors reaches $\frac{\pi}{2}$ rad/s. In which case the mechanical torque is equal to,

$$T_{\mathrm{me}}^{\mathrm{max}} = \frac{3}{2}\, p\, |\vec{i}|^{\mathrm{ref}}\, \psi_{\mathrm{PM}} \tag{4.11}$$

Once the load torque exceeds the pull-out torque of the motor, it will lose synchronism with the current vector and stall.

In the following sub sections, current control will be discussed according to the tool-chain structure shown in figure 2.43, hence covering Texas Instruments and TI-Micro hardware examples.

4.2.1 Current & Frequency Control-TI

Purpose of this section is to realize current & frequency control of a PMsyn or BLDC machine using Texas Instruments hardware. The tool chain as introduced in figure 2.45 is used as a template for the ensuing subsections. However, additional HIL examples will be provided. Note also, that in many field-oriented control examples (to be discussed later in ensuing sections), open loop current & frequency control is often added as a first operational check.

4.2.1.1 Current & Frequency control-F0049-GANINV: SIM

Reference file: `C&F_0049_GANINV-SIM.vsm`

This example considers the simulation of a PMsyn Teknic motor [2] connected to a Texas Instruments F0049 GANINV converter, as shown in figure 4.123. The aim is to examine operation of the drive using a fixed point controller, that will subsequently be used in the HIL implementation.

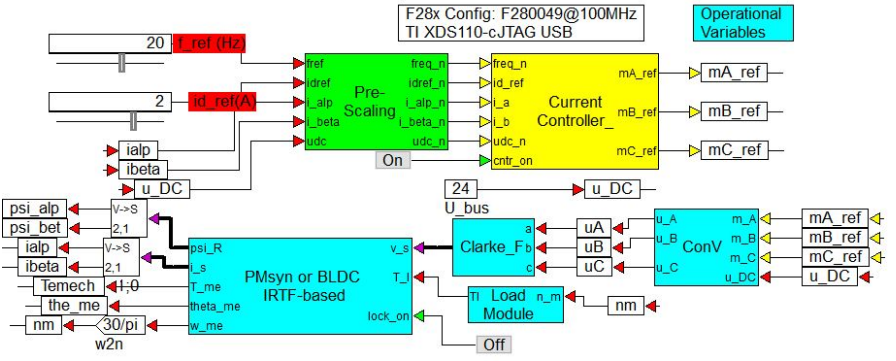


Fig. 4.123 Current & Frequency control: simulation

Drilling into the 'current controller' reveals the structure shown in figure 4.124. Outputs of the controller are the three modulation indices that control the three converter half bridges. Inputs are the per unit direct axis reference current `ampld_n`, and per unit frequency `freq_n`. For all controllers considered the aim is to generate the modulation indices that are required by the PWM units. Consequently, it is important to convert the current controller reference voltages to modulation indices using expression (4.12).

$$m^{\text{ref}} = \frac{2\,u^{\text{ref}}}{u_{\text{DC}}} \tag{4.12}$$

which requires access to the measured DC bus voltage u_{DC}, shown as `Vdcn` (which is a per unit variable) in figure 4.124.
Moving one level into the `Syn Current controller` module leads to the structure shown in figure 4.125. An overview of 'synchronous current control' principles is given in section 2.1.4 and these principles are applied here using

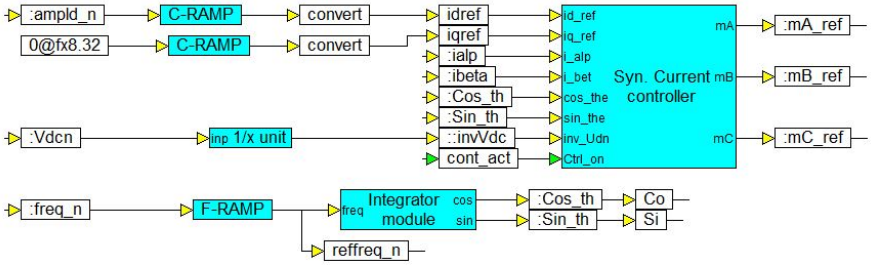


Fig. 4.124 Current controller structure

the modules shown. The per unit measured currents in (α, β) coordinates `i_alp`, `i_bet` are transformed to (d, q) coordinates using a 'Forward Park' module `Park-fwd`.

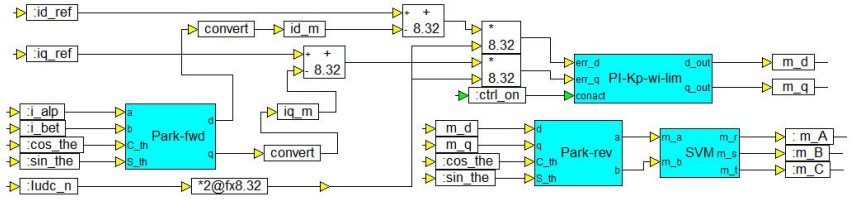


Fig. 4.125 One level into 'Synchronous Current controller'

The per unit inverted DC bus voltage shown in figure 4.125 as `Iudc_n`, is used to convert the PI controller (module `PI-kp-wi-lim`) outputs to modulation indices `m_d`, `m_q`. A reverse Park transformation is then used to arrive at stationary, (α, β) indices. Both proportional gain, bandwidth and the maximum modulation index of the PI controller must be provided in the 'operational variables' dialogue box. The space vector modulation 'SVM' unit centers the active vectors, relative to the sampling interval, as discussed in section 2.1.5. The voltages supplied to the motor `u_A_ref`,`u_B_ref`,`u_C_ref` (see figure 4.123) are found by multiplying the modulation indices `mA_ref`, `mB_ref`, `mC_ref` by half the DC bus voltage as executed in module `ConV`. These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables. The aim of the switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sub chapter 2.1.5). A quadratic load is introduced, which provides a value of 100 mNm at a shaft speed of 1000 rpm. The IRTF based PM model [13] has a user option to select a synchronous or BLDC type machine, as discussed earlier (see figure 2.12). When selecting the 'BLDC' option a sixth harmonic component is added to the PM flux locus. In this case the 'PMsyn' option is selected, given the machine used for this

example.

The results shown in figure 4.126, confirm the presence of current and frequency control. Note that at start up ramp functions are used, which determine the rate of reference current and frequency variable changes, as set in the 'Operational Variables' dialogue box.

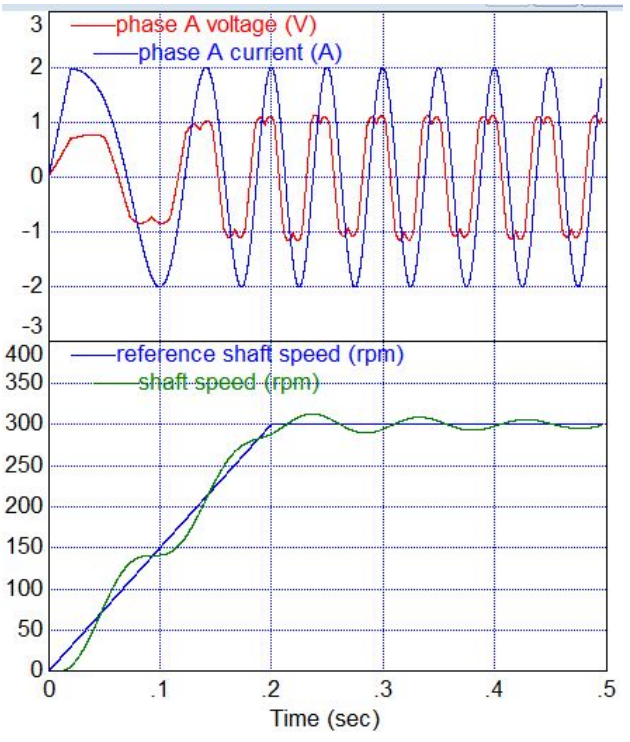


Fig. 4.126 SIM result, PMsyn operation, current & frequency control

Of importance is to consider the vector plot, see figure 4.127 at time mark: 131 ms, which shows the orientation of the unity d-axis ('green'), voltage vector ('blue') and current vector ('red').
Observation of the current vector confirms that it is indeed aligned with the d-axis and its amplitude is 2 A, as expected because `iq_ref` is set to zero. Note also that the voltage vector leads the current vector, which is expected due to the inductive nature of the machine. Furthermore, the PM flux vector will be aligned with the current vector, under no-load conditions.

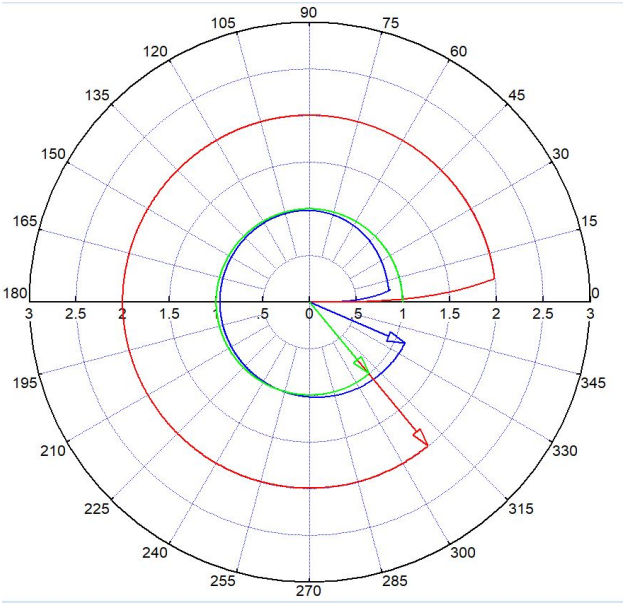


Fig. 4.127 SIM result, PMsyn operation, current & frequency control: vector plot

4.2.1.2 Current & Frequency control-F0049-GANINV: PIL

Reference files:

- `C&F_F0049-GANINV-PIL.vsm`
- `C&F_PIL_mod.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F280049. This implies that the controller module shown in the simulation (see figure 4.123) is now replaced with a file: `C&F_PIL_mod.out`, as indicated in figure 4.128.

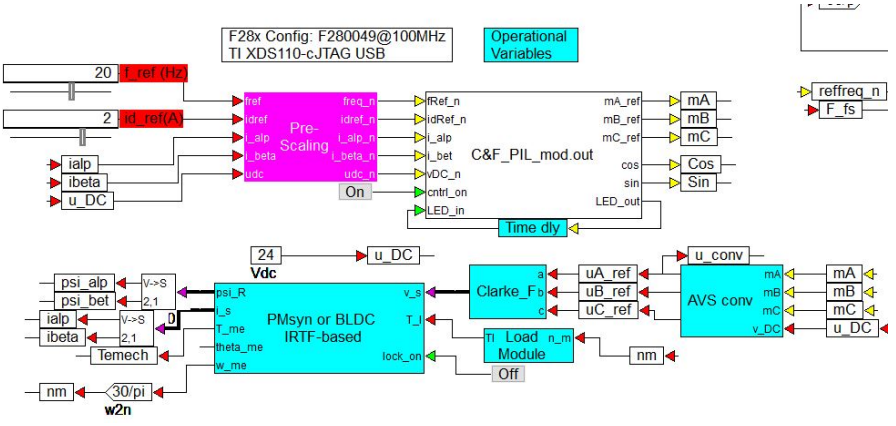


Fig. 4.128 PIL based drive, PMsyn operation, current & frequency control

The target interface model that executes the out-file, generates the modulation indices required for the converter motor and these components are not affected by the use of the PIL. Generation of the out file is based on the PIL module (see figure 4.129), which must be compiled using the command: `toolbox>codegen`.

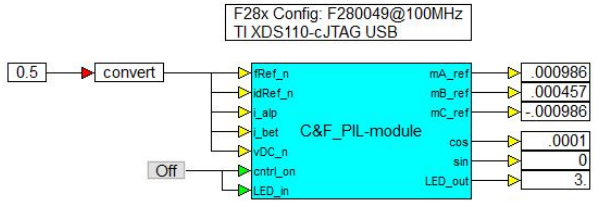


Fig. 4.129 PIL module, current & frequency control

Moving one level into the PIL module `C&F_PIL_module` (see figure 4.130), shows the current controller as introduced in section 4.2.1.1.

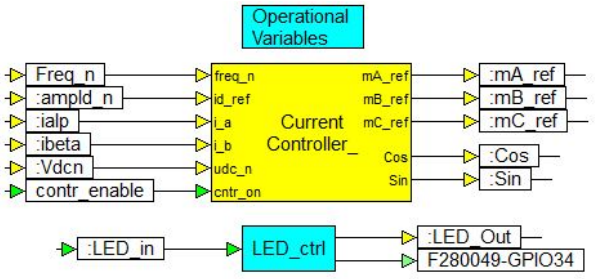


Fig. 4.130 Content PIL module, current & frequency control

Added to the current controller is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that PIL control is operational. In addition, two variables `Cos`, `Sim` are provided as outputs and these are used in a vector plot to show the 'd' axis of the controller algorithm.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational. Results produced by PIL operation are identical to the plots shown in figures 4.126, 4.127 and are therefore not shown here.

4.2.1.3 Current & Frequency control-F0049-GANINV:COSIM

Reference files:

- `C&F_0049_GANINV-COSIM.vsm`
- `GAN_INVv12.psimsch`
- `my motor model_v3.psimsch`

To evaluate operation of a controller with a switching converter, in this case the `GANINV`, use can be made of a `PSIMCoupler` module, as indicated in figure 4.131. This approach implies the use of PSIM together with Embed, which is referred to as 'co-simulation'.
The `Current Controller` module is hatched in this case, as its sampling frequency is locally set to 10 kHz, whereas the 'system' frequency (set in the 'system properties' dialogue box) is set to 20 MHz. The latter is much higher, given the need to accurately represent 20 kHz voltage switching wave-

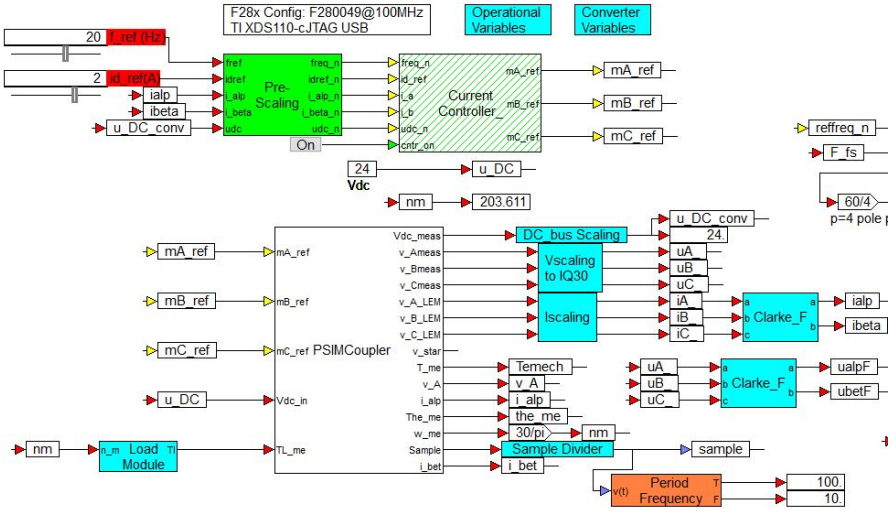


Fig. 4.131 COSIM based drive, current & frequency control

forms generated by the PWM unit of the converter. The content of the `Current controller` module was introduced earlier (see figure 4.124). However, the current and voltage measurements now make use of the signal processing circuitry connected to the half-bridges.

In this case the current transducers are in the converter phases, as shown earlier (see figure 4.10). After sampling and signal-processing in the converter, these variables appear as: `iA_`, `iB_`, `iC_` in figure 4.131. The low pass filtered voltages (which removes the PWM component) of the three half-bridges, is also present in figure 4.131 and are also scaled and sampled by the module `Vscaling to IQ30`. In this case the per unit variables are rescaled to actual values:`uA_`, `uB_` and `uC_` respectively.

The 'Converter Variables' dialogue box, as shown in figure 4.131, defines the scaling used by the Analog to Digital converter, which has a maximum input value of 3.3 V and is directly linked to the signal processing undertaken for voltages and currents as discussed above.
For example, the 'ADC-fullscale-voltage' value is the theoretical peak-to peak input voltage, which must be applied to the low-pass filter network (see figure 4.12) to ensure that the ADC input level varies between $0 \rightarrow 3.3$V. The current transducer voltages: `v_A_LEM`, `v_B_LEM` and `v_C_LEM`, are sampled by the `Iscaling` module shown in figure 4.131. When dealing with the actual MCU, the ADC converter also takes over this sampling process.
Apart from the measured voltage/currents generated by the converter, the PSIM coupler unit also provides the actual motor model current vector: `i_alp`,

`i_bet` (in stationary coordinates), phase 'A' voltage and speed, which are presented by the variables `v_A` and `n_m` (after conversion to rpm) respectively (see figure 4.9).

Results achieved with the COSIM approach are shown in figure 4.132.

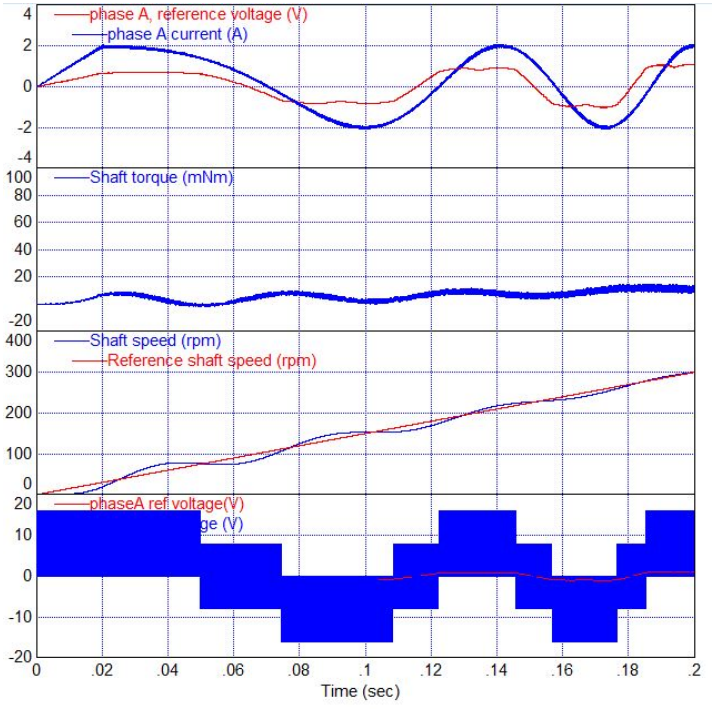


Fig. 4.132 Results using COSIM, current & frequency control of a PMsyn drive.

Indicated in said figure is the phase 'A' voltage waveform together with the reference voltage, phase current, shaft torque and shaft speed. A vector plot, given in figure 4.133, at time 126 ms after starting the co-simulation, shows the measured LPF voltage vector ('blue'), unity d-axis vector ('green') and the actual (from the IRTF motor model) current vector ('red'). Note that the impact of switching is clearly reflected in the current vector locus, e.g. changes in end-point direction due to use of 'active' or 'zero' voltage vectors [13]. Co-simulation, as discussed in this sub-section, is instructive for understanding the impact of using a switching converter on the drive operation.

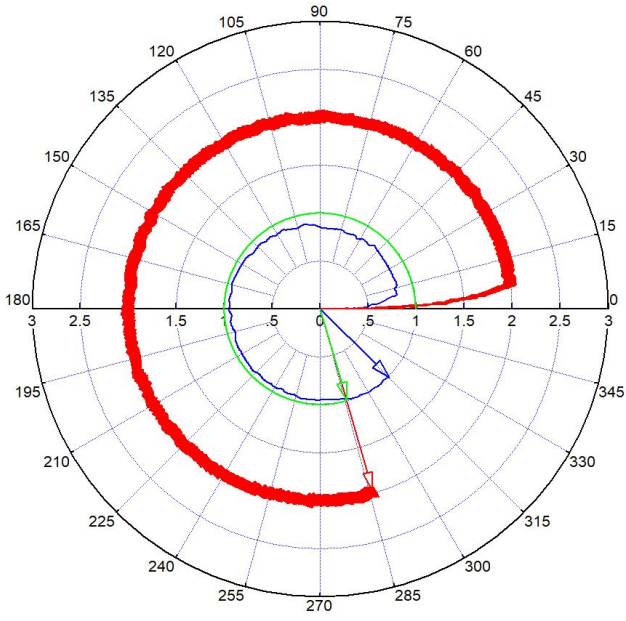


Fig. 4.133 Results using COSIM, current & frequency control of a PMsyn drive: vector plot.

4.2.1.4 Current & Frequency control-F0049-GANINV: HIL

Reference files:

- `CandF_Gal_280049-compile.vsm`, for HIL-compile
- `CandF_Gal_280049-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), shown in figure 4.134, represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F280049` module), which is connected to a PMsyn Teknic motor[2].

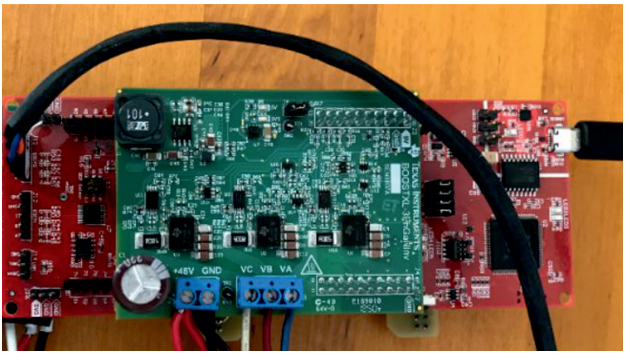


Fig. 4.134 F280049 MCU-BOOSTXL-GANINV setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects need to be considered such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches.
A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration.

The model given in figure 4.135 shows the 'current controller' module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct axis reference current `id`, and the frequency `Freq`, which are converted to 'per unit' values in the 'pre-scaling' module. Note that the quadrature reference current is set to zero in this example.
Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Output of the Controller module is the per unit DC bus voltage `udc_n`, which

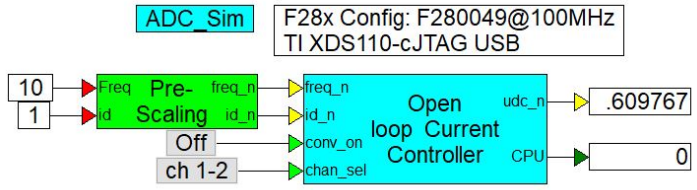


Fig. 4.135 HIL-Compile model of a PMsyn drive with current & frequency control

must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts. In addition, a `CPU` output is added, which provides the user with a % reading of MCU activity, relative the controller sample time.
Also shown in figure 4.135 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. These values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes shown in figure 4.136.

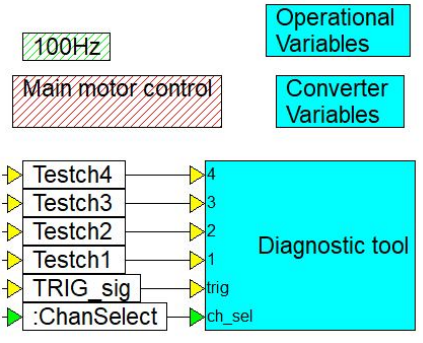


Fig. 4.136 One level down into the 'Open loop Current Controller' module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.137), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current-frequency.
The 'Converter Variables' dialogue box, as given in figure 4.138, defines the scaling used by the Analogue to Digital converter, as discussed earlier (see

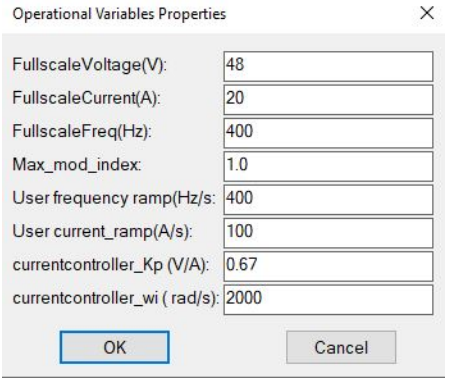


Fig. 4.137 HIL: Dialogue box 'Operational Variables'

equation (4.2)).

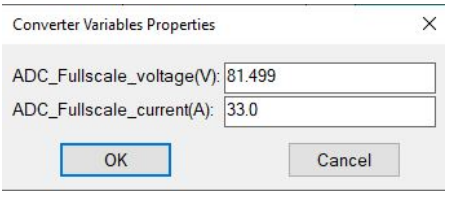


Fig. 4.138 HIL: Dialogue box 'Converter Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using the 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.136 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 4.139. Of these, the 'Current controller' has been discussed earlier (see figure 4.125), where it was used in development tool-chain elements: SIM, PIL

and COSIM.

The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F280049` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after every second (for this drive) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

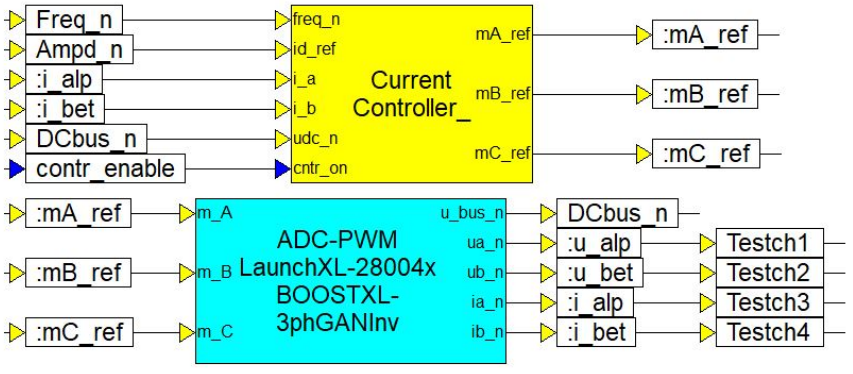


Fig. 4.139 One level down into the 'Main motor controller' module

The `ADC-PWM-LaunchXL-28004x BOOSTXL-3phGANINV` unit is used generate the PWM signals for the converter. In addition it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively.

ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offsets is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage, `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 4.140 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).

Central to this figure is the Embed 'target execution module' , which executes said out-file. Two sliders are used to set the direct axis current reference value and the electrical frequency . Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

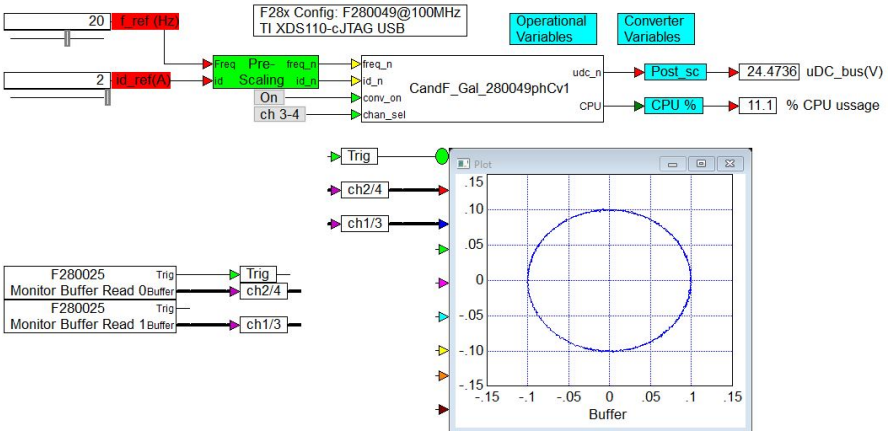


Fig. 4.140 HIL-Debug model, F0049-GANINV of a PMsyn drive with current & frequency control

The plot given in figure 4.140 displays the per unit LPF measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit voltage vector locus. Note that the current reference slider `id_ref` is set to 2.0 A (and `iq_ref`=0), hence the per unit current amplitude should be $|\vec{i}|_{\text{ref}} = {}^{2}/_{20} = 0.1$, given that a full-scale current of i_{fs}=20 A is used. The electrical frequency reference in figure 4.140 is set to 20 Hz, which corresponds to a shaft speed equal to $n_m = {}^{60\, f_{ref}}/_{p} = 300$ rpm, given that a four pole pair $p = 4$ machine is used.

For comparative purposes an oscilloscope [20] with a DC current probe was used to measure the phase 'A' current, as shown in figure 4.141.
The measured current amplitude is in agreement with the theoretical models shown earlier, e.g the Cosim example (see figure 4.132), which has the same current/frequency reference settings as used in this HIL example.

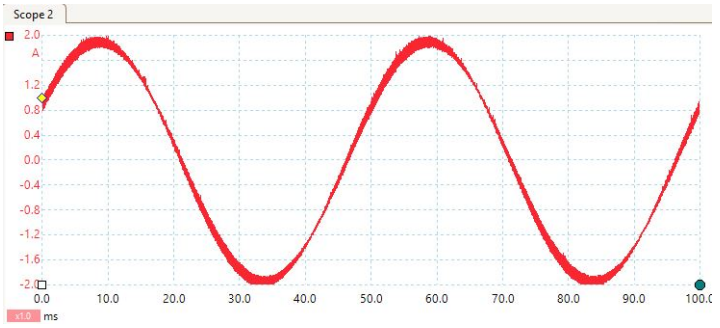


Fig. 4.141 Experimental results: measured phase current, phase 'A'

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.2.1.5 Current & Frequency control-Turnigy-F069-boostXL8301: HIL

Reference files:

- `C&F_069_Turnigi_boost_8301-compile.vsm`, for HIL-compile
- `C&F_069_Turnigi_boost_8301-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), as shown in figure 4.142 represents the `BOOSTXL-8301` converter (located on the `LAUNCHXL-F28069` module), which is connected to a 'Turnigy' Brushless DC motor.

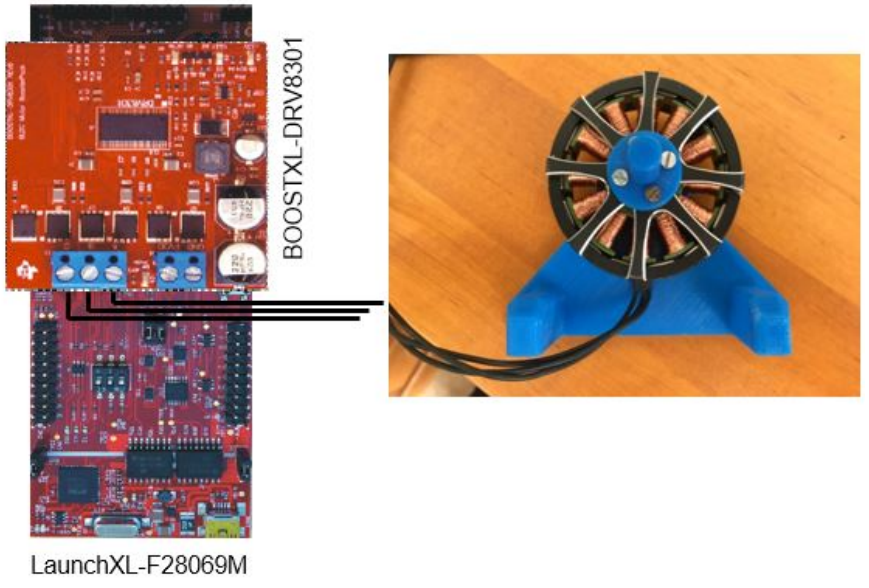


Fig. 4.142 F28069 MCU-BOOSTXL-8301 converter setup for HIL operation

A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration.
The model given in figure 4.143 shows the `Open loop current controller` module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct axis reference current `id`, electrical frequency reference `freq`, which need to be converted to 'per unit' values in the 'pre-scaling' module.
Two logic inputs are provided, which are used to turn on the converter and to select which combination of diagnostic channels should be viewed. Outputs of the Controller module are the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a

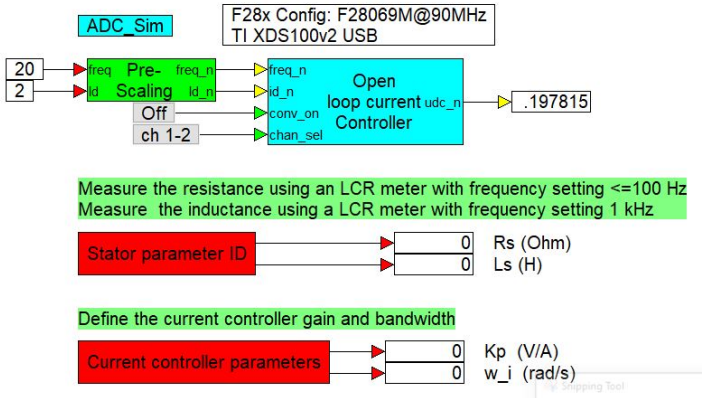


Fig. 4.143 HIL-Compile model of a BLDC drive with Current & Frequency control

reading in Volts.
Also shown in figure 4.143 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 4.144.

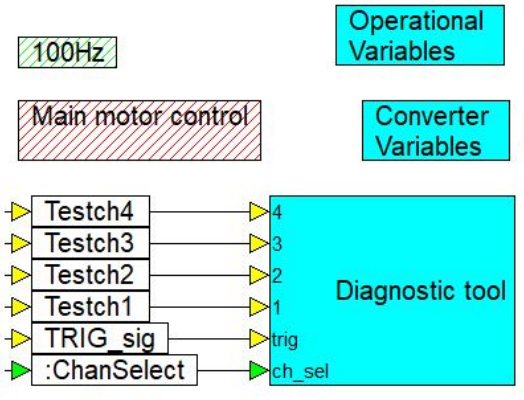


Fig. 4.144 One level down into the 'Open loop Current Controller' module

In this case two dialogue boxes are used, as shown in figure 4.144, which

contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (figure 4.145), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference frequency and current.

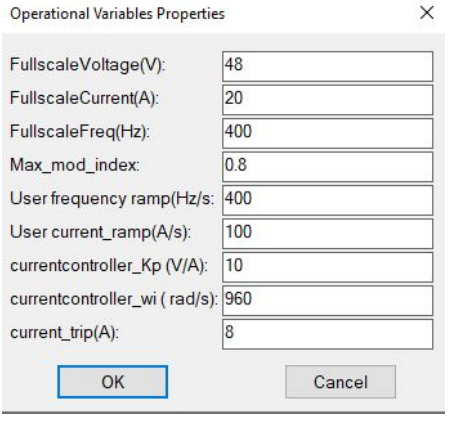


Fig. 4.145 HIL: Dialogue box 'Operational Variables'

In addition, the current controller gain/bandwidth are assigned here together with the chosen duty cycle (therefore, the upper absolute voltage limit, set to $0.8 * 12 = 9.6$V).

The 'Converter Variables' dialogue box, as given in figure 4.146, defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.4).

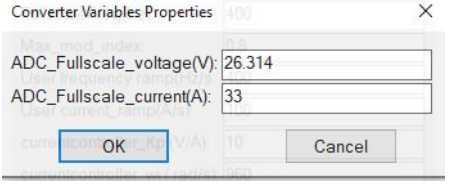


Fig. 4.146 HIL: Dialogue box 'Converter Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph when operating in HIL-debug. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.144 is the `Main motor control` module which contains the main control algorithm as needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 4.147. Of these, the `Current Controller_` module has been discussed previously, given that it is the module used in phase : SIM of the development tool-chain (in this case for a different MCU, see figure 4.124).
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module.
This module is configured for with the following basic settings :

- PWM frequency: 15 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: 0.11 μs
- Board in use: `LAUNCHXL-F28069` with `BOOSTXL-8301` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which ensures that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed. The converter dead-time for the PWM is set to 10 MCU clock cycles. Outputs from said unit are the per unit measured DC bus voltage `DCbus_n`, low-pass filtered half-bridge voltages, forward Clarke transformed to stationary coordinates: `u_alp`, `u_bet` and forward Clarke transformed motor currents `i_alp`, `i_bet` respectively.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 4.148 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. Two sliders are used to set the current reference amplitude `id_ref` and frequency `f_ref`. A post-scaling module is used to convert the per-unit measured DC bus to actual voltage, as shown with a numeric display.

Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.
The plot given in figure 4.148 displays the per unit motor current vector locus. This requires the channel select button to be set to `ch 1-2`. In this case the frequency reference slider is set to 20 Hz , hence the shaft speed for a 7 pole pair ($p = 7$) machine is equal to $n_m = {}^{60\, f_{\text{ref}}}/_{p} = 171$ rpm.

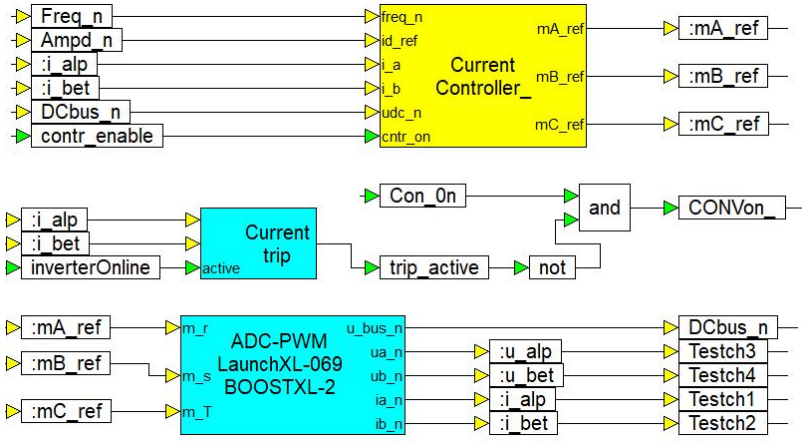


Fig. 4.147 One level down into the 'Main motor controller' module

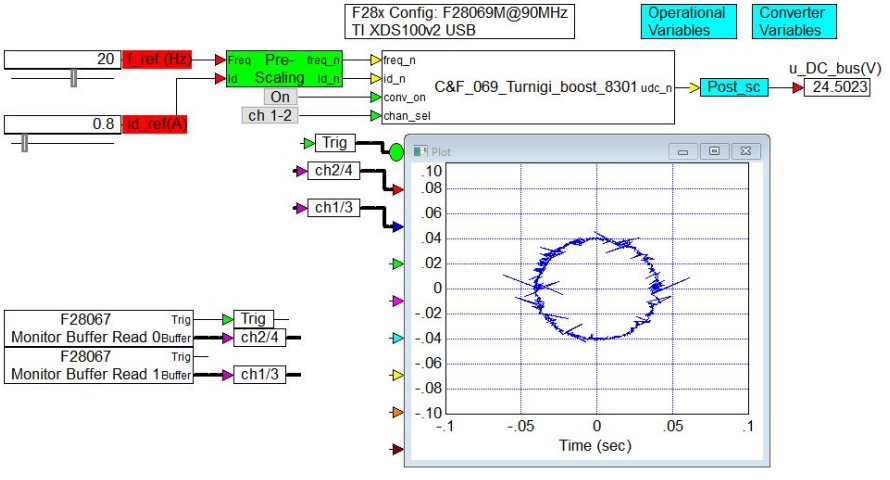


Fig. 4.148 HIL-Debug (run) model of a BLDC drive with current &frequency control

For comparative purposes an oscilloscope [20] with a DC current probe was used to measure the phase 'A' current, as shown in figure 4.149.
The measured current amplitude is in agreement with the vector plot which shows the per unit value $^{0.8}/_{20} = 0.04$ (see figure 4.148) and this confirms that the ADC-fullscale current setting in the dialogue box (figure 4.146) is indeed correct.

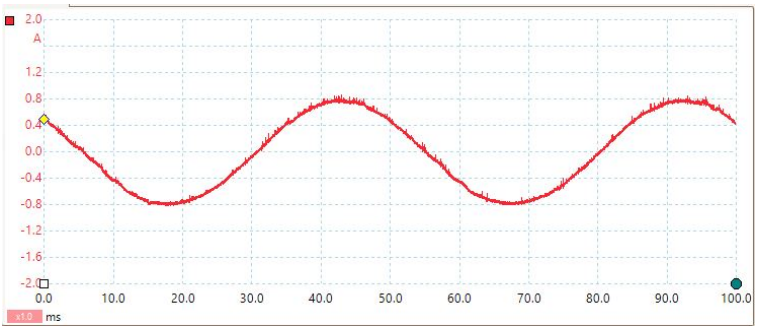


Fig. 4.149 Experimental results: measured phase current, phase 'A'

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC bus voltage level established, turn on the converter (using the `on` button, shown) and monitor machine operation.

Post Analysis

Apart from verification of drive operation under C&F control, it is helpful to use the results for estimating key parameters, such as the pole pair number p and PM rotor flux amplitude ψ_{PM}. For this purpose it is useful to consider figure 4.150, that shows the voltage vector locus when operation under C& F

control AND no-load operation. This implies that the PM flux and current vector must be in phase (ignoring any mechanical friction torque effects).

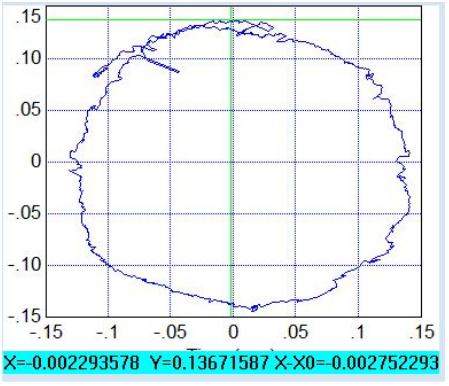


Fig. 4.150 Vector plot Current & Frequency control, voltage locus: no-load operation

On the basis of this diagram and the current vector locus (see figure 4.148), the EMF vector amplitude e can be estimated using Pythagoras namely,

$$e = \sqrt{v_s^2 - (i_s\, R_s)^2} - \omega_s\, L_s\, i_s \tag{4.13}$$

where L_s, R_s are the estimated motor inductance and resistance respectively, which are presumed to have been measured previously using a `L-C-R` meter (as required for setting the gain and bandwidth of the current controller, see equation (2.32)). A vector plot showing the vectors that represent the EMF, voltage across the inductance and voltage across the resistance is shown in figure 2.13 for the case $\vec{i} = ji$. Note that in this example the current reference is set to $\vec{i} = i$, hence the vector diagram needs to be seen in the light of this change, i.e the current and flux vectors will now be in phase (assuming no mechanical load).
The frequency $\omega_s = 2\,\pi\, f^{\text{ref}}$ is defined by the user, like the current i_s. Once the EMF has been found the rotor flux is calculated using $\psi_{\text{PM}} = {}^{e}/_{\omega_s}$.
For the machine in question the voltage amplitude is calculated from figure 4.150, which gives (after correction of the per unit value) $u_s \approx 6.5\text{V}$. Subsequent use of equation (4.13) gives a EMF of $e \approx 3.5\text{V}$ and rotor flux of $\psi_{\text{PM}} \approx 28\text{mWb}$ for the motor in question.

Identification of the pole pair number p can in some cases, (like 'out-runner' type BLDC machines, as considered in the example) be found by simply

counting the number of rotor magnets and dividing by 2. Alternatively the rotational shaft speed (rpm/s) can be used, by considering that the latter is defined as

$$n_{\text{me}} = \frac{f^{\text{ref}}}{p}(\text{rpm/s}) \tag{4.14}$$

Hence by setting the user shaft speed to one rotation per second (by observation of the shaft) by adjusting the reference frequency leads to the pole pair number. For example, for a seven pole pair machine $p = 7$ (as used here), a reference frequency of 7 Hz is needed to realise a $n_{\text{me}} = 1$ rpm/s. An alternative approach is to set the reference frequency to zero and apply sufficient current to ensure a stable rotor position and one where the user can rotate the shaft by hand. Consider a time clock, with 12 o'clock, being aligned with the initial position of the rotor. Then a machine with, for example 4 pole pairs, will encounter stable positions at: 3, 6, 9, 12 o'clock, whenever the user rotates the shaft manually.

4.2.2 Current & Frequency Control-STM

The purpose of this section is to realize current & frequency control of a PMsyn or BLDC machine using ST Micro hardware. The tool chain as introduced in figure 2.45, is used as a template for the following subsections. Note that in many field-oriented control examples (to be discussed later in ensuing sections) open loop current & frequency control is often added as a first operational check.

4.2.2.1 Current & Frequency control-F303-IHM07: SIM

Reference file: `C&F_F303_IHM07-SIM.vsm`

This example considers the simulation of a BLDC out-runner machine [31] connected to a ST Micro F303 IHM07 converter, shown in figure 4.151. This motor has a EMF threshold speed (see equation (2.12a)) of $n_e = 2250$ rpm, hence relatively high. The aim is to examine operation of the drive using a fixed point controller, that will subsequently be used in the HIL implementation.

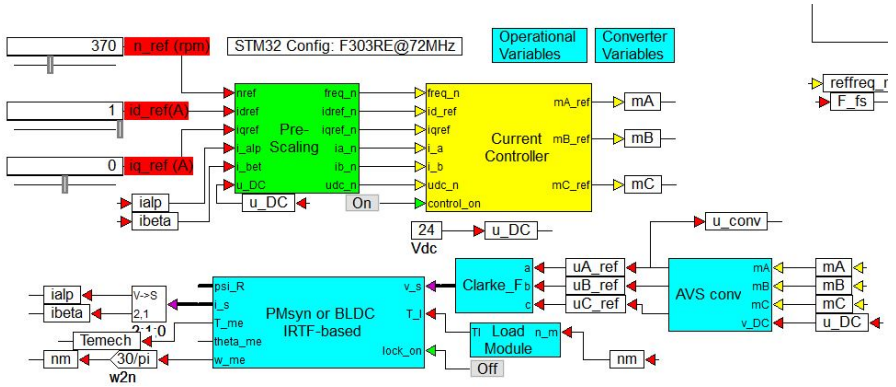


Fig. 4.151 Current & Frequency control: simulation

This example makes use of the 'current controller' introduced earlier (for a different MCU, see figure 4.124). Note that in this example, the user is able to control the reference direct axis AND quadrature axis current components. Outputs of the controller are modulation indices that control the three converter half bridges.
The voltages supplied to the motor `uA_ref`,`uB_ref`,`uC_ref` are found by multiplying the modulation indices `mA`, `mB`, `mC` by half the DC bus voltage as executed in module `AVS conv` (see figure 4.151). These half-bridge voltages are in effect, the 'average voltage per sample t_k' phase variables. The aim of the switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sub chapter 2.1.5). A quadratic load is introduced, which provides a value of 1 mNm at a shaft speed of 1000 rpm.
The IRTF based PM model [13] has a user option, which allows use of a synchronous or BLDC type machine, as discussed earlier (see figure 2.12). When selecting the 'BLDC' option, a sixth harmonic component is added to the PM flux locus, as is required for this example.

The results shown in figure 4.152, confirm the presence of current and frequency control. Note that at start up ramp functions are used, which determine the rate of reference current and frequency variable changes, as set in the 'Operational Variables' dialogue box. Note that the voltage levels at the speed shown in this result are low, as this machine is designed for high speed operation, e.g for model aircraft propulsion.

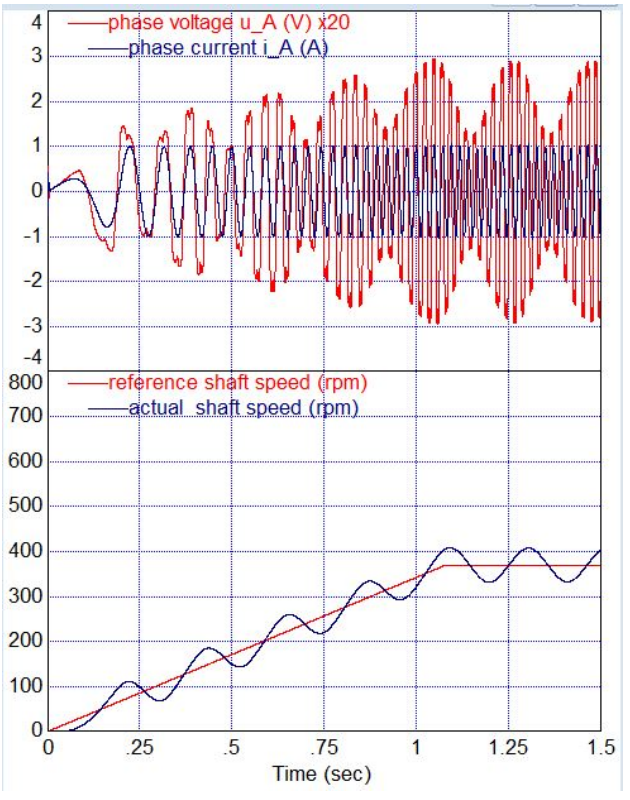


Fig. 4.152 SIM result, PM syn operation, current & frequency control

4.2.2.2 Current & Frequency control-F303-IHM07: PIL

Reference files:

- `C&F_F303_IHM07-PIL.vsm`
- `C&F_PIL_mod-ST.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F303RE. This implies that the controller module shown in the simulation (see figure 4.151) is now replaced with a target interface module, with file: `C&F_PIL_mod_ST`, as indicated in figure 4.153.

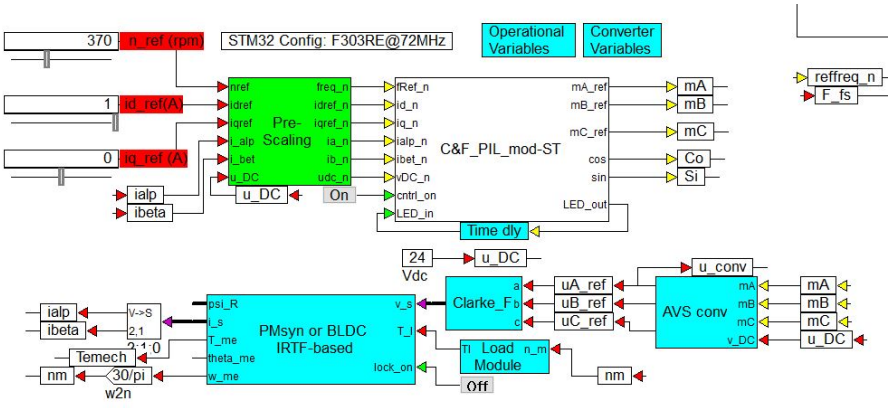


Fig. 4.153 PIL based drive, PM syn operation, current & frequency control

The target interface model that executes the out-file, generates the modulation indices required for the converter motor and these components are not affected by the use of the PIL. Generation of the out file is based on the PIL module (see figure 4.154), which must be compiled using the command: `toolbox>codegen`.
Moving one level into the PIL module `C&F_PIL_mod` shows the current controller as introduced in subsection 4.2.1.1.
Added to the current controller is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that the PIL control is operational. In addition, two variables `Cos`, `Sim` are provided as outputs and these are used in a vector plot to show the 'd' axis of the controller algorithm.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Nucleo unit, which indicates that the MCU is operational. Results produced by PIL operation are identical to the plot shown in figure 4.152 and are therefore not shown here.

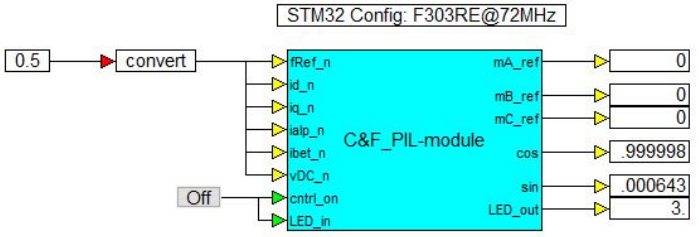


Fig. 4.154 STM PIL module, current & frequency control

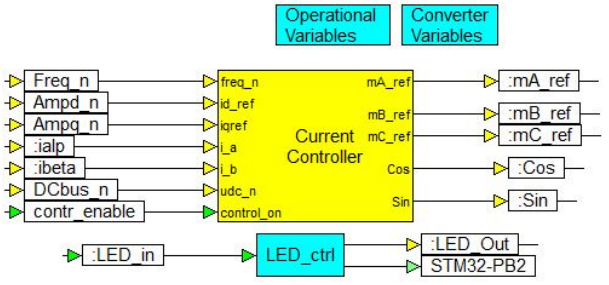


Fig. 4.155 Content STM PIL module, current & frequency control

4.2.2.3 Current & Frequency control-F303-IHM07:COSIM

Reference files:

- `C&F_F303_IHM07-COSIM.vsm`
- `IHM07_BLDC_mot.psimsch`
- `my motor model_v3.psimsch`

To evaluate operation of a controller with a switching converter, in this case the `NUCLEO` unit, use can be made of a `PSIMCoupler` module, as indicated in figure 4.156. This approach implies the use of PSIM together with Embed, which is referred to as 'co-simulation'. The `Current Controller` module , hatched in this case as its sampling frequency is locally set to 10 kHz, whereas the 'system' frequency (set in the 'system properties' dialogue box) is set to 20 MHz. The latter is much higher, given the need to accurately represent 40 kHz voltage switching waveforms generated by the PWM unit of the converter. The content of the `Current controller` module was introduced

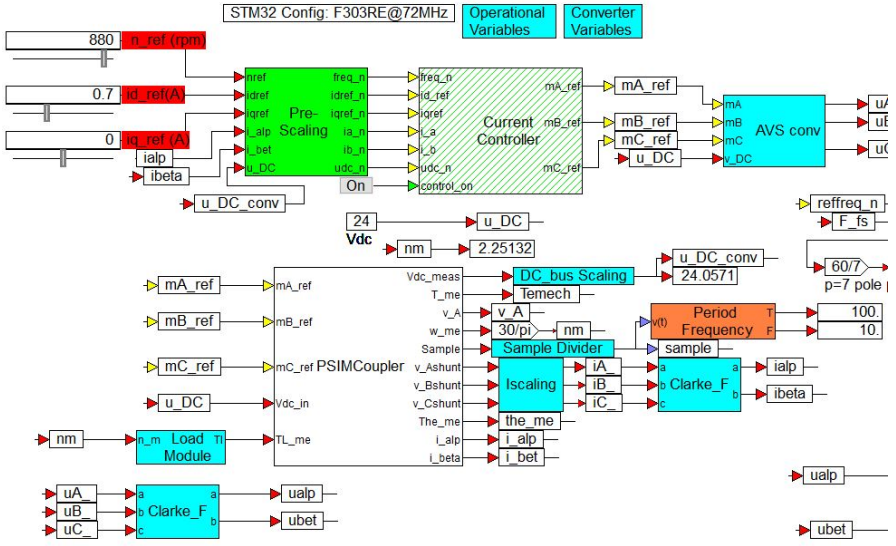


Fig. 4.156 COSIM based drive, current & frequency control

earlier (see figure 4.3). However, the current measurements now make use of the signal processing circuitry connected to the half-bridges.
In this case, the current shunts are connected to the lower converter switches, as shown in figure 4.157. After sampling and signal-processing in the converter, these variables appear as: `iA_`,`iB_`,`iC_` in figure 4.156. Note that the signal processing for the 'C' half-bridge is not shown in figure 4.157.
Further observation of figure 4.157 shows the presence of the module `PMSM`, which is a generic IRTF model of the IPM synchronous machine machine, as presented earlier (see figure 2.17). However, the PSIM model, also introduced earlier, see figure 4.11, has an 'electrical interface' to allow use of said model with a converter.
A PWM unit referred to as `3-ph PWM`, generates the gate control signals used to switch on/off the upper or lower devices of each converter leg.

An important part of the data acquisition process is measurement of the phase current with the aid of 0.33Ω shunt resistors (see figure 4.157). Furthermore, the current shunt voltage is amplified by an op-amp circuit with gain G_{sh}, hence the voltage output `v_Ashunt` for the MCU (and an output of the PSIM module) is equal to:

$$v_{\text{shunt}} = i_{\text{conv}}\, R_{\text{RC}}\, G_{\text{sh}} \tag{4.15}$$

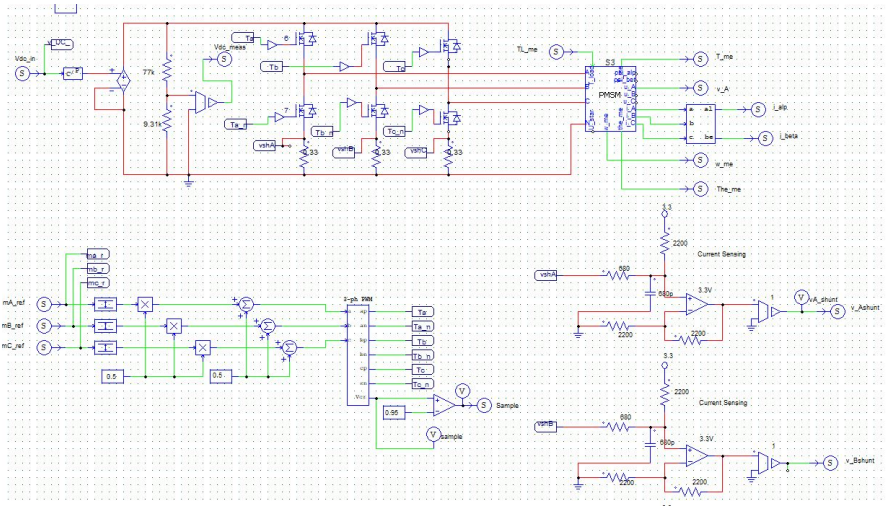


Fig. 4.157 Content PSIM coupler module, current& frequency control

where R_{RC} represents the 0.33Ω shunt resistor.
The 'Converter Variables' dialogue box (see figure 4.156), defines the scaling used by the Analog to Digital converter, which has a maximum input value of 3.3 V. It is directly linked to the signal processing undertaken for voltages and currents as discussed above. For example, the 'ADC-fullscale-voltage' value is the theoretical DC bus voltage that must be applied to the attenuation/op-amp circuit with output `Vdc_meas`, to ensure that the ADC input level equals 3.3 V.
These fullscale variables are defined as:

$$\mathrm{ADC-fullscalevoltage} = \frac{3.3}{G_{\mathrm{R}}} \tag{4.16a}$$

$$\mathrm{ADC-fullscalecurrent} = \frac{3.3}{R_{\mathrm{RC}}\, G_{\mathrm{sh}}} \tag{4.16b}$$

where G_{R}, is the attenuation used for measuring the DC bus voltage. Similarly, the 'ADC-fullscale-current' value is the peak to peak current in the shunt resistance, which corresponds to an ADC input voltage range of $0 \rightarrow 3.3$ V. Hence the ADC-fullscale current value is determined by the shunt resistance value R_{RC} and the signal amplification gain G_{sh}, between said resistor and ADC input. The voltage and current measurements are unipolar and bipolar

respectively. This implies that the voltage measurement is referenced to the negative converter DC bus terminal, whilst the current is referenced to the center of the ADC voltage span, i.e $^{3.3}/_{2}$ V, as provided by an op-amp.
Note that the ADC dialogue values, are board dependent and need to be found by inspection of the circuit (or provided). The current shunt voltages: `v_Ashunt`,`v_Bshunt` and `v_Cshunt`, are sampled by the `Iscaling` module shown in figure 4.156. When dealing with the actual MCU, the ADC converter also takes over this sampling process. Apart from the measured voltage/currents generated by the converter, the PSIM coupler unit also provides the actual motor model current vector (in stationary coordinates), phase 'A' voltage and speed, which are represented by variables: `v_A`, `i_alp`,`i_bet` and `n_m` (after conversion to rpm) respectively (see figure 4.156).

Results achieved with the COSIM approach are shown in figure 4.158.

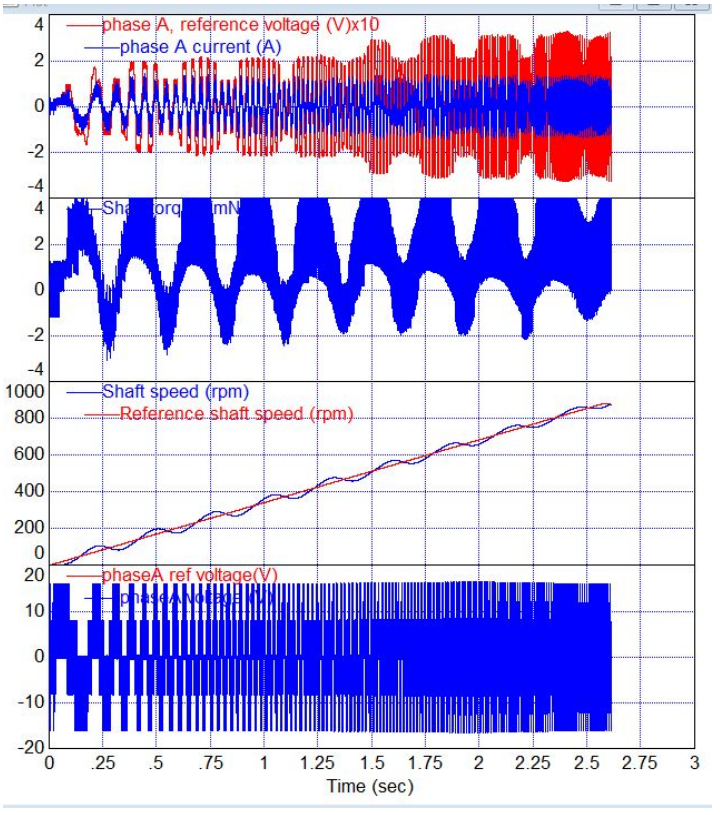


Fig. 4.158 Result COSIM, current & frequency control PMsyn drive

Indicated in said figure is the phase 'A' voltage waveform together with the

reference voltage, phase current, shaft torque and shaft speed. Co-simulation, as discussed in this sub-section, is instructive for understanding the impact of using a switching converter on drive operation, notably the phase current and shaft torque ripple.

4.2.2.4 Current & Frequency control-F303-IHM07: HIL

Reference files:

- `C&F_STM_F303RE_IHM07-compile.vsm`, for HIL-compile
- `C&F_STM_F303RE_IHM07-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), shown in figure 4.159 shows the `IHM07-NUCLEO` module, which is connected to a BLDC motor[31].

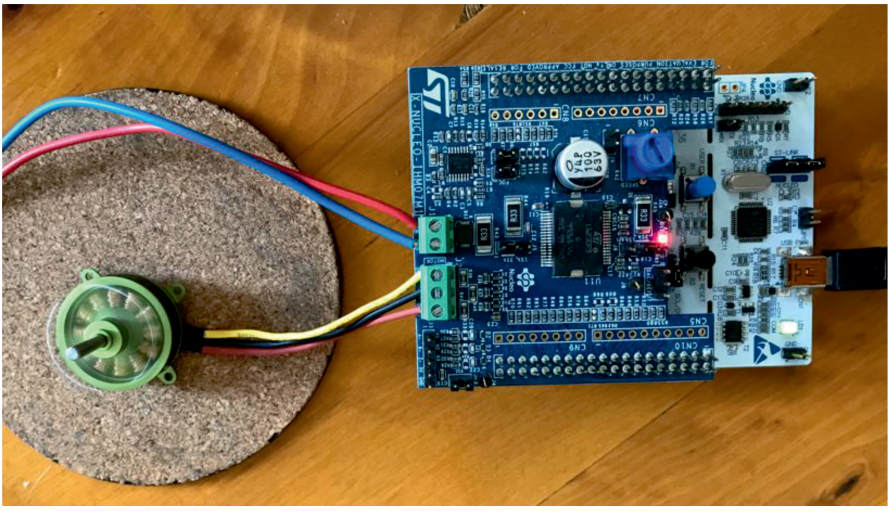


Fig. 4.159 F303RE MCU-IHM07 Nucleo converter setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects that need to be considered, include allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.160, shows the current controller: `C&F` module, which must be compiled to generate an

out-file.

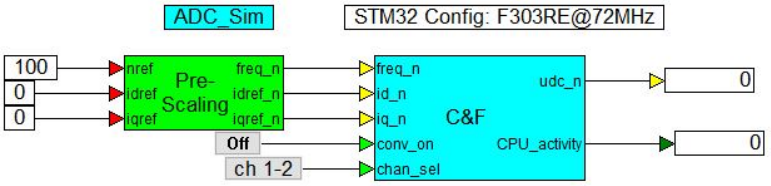


Fig. 4.160 HIL-Compile model of a BLDC drive with current & frequency control

Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct/quadrature reference currents `idref`, `iqref` and the shaft speed reference `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Output of the Controller module is the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts. Furthermore, a `CPU` output is added, which provides the user with a % reading of MCU activity, relative the controller sample time.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes shown in figure 4.161.

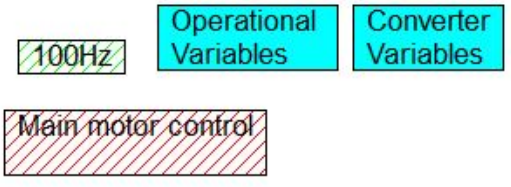


Fig. 4.161 One level down into the Current Controller module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.162) is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference currents and frequency.

The 'Converter Variables' dialogue box (see figure 4.163), defines the scaling used by the Analogue to Digital converter, as discussed earlier (see equation (4.16)). Furthermore, for convenience, the BLDC parameters of the motor used in this example are provided.

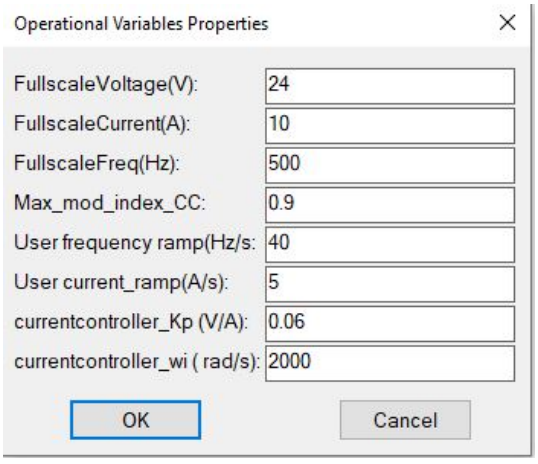


Fig. 4.162 HIL: Dialogue box 'Operational Variables'

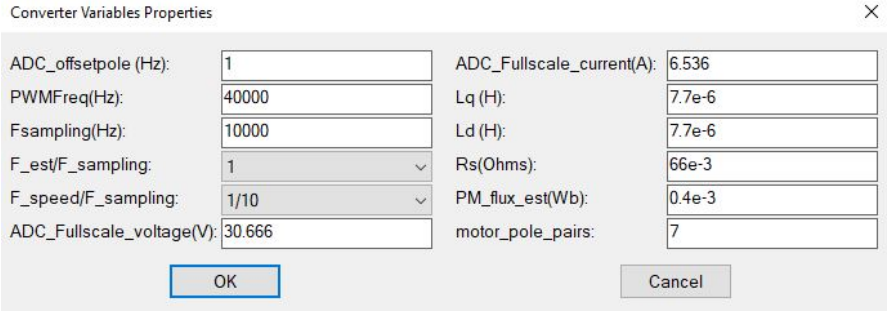


Fig. 4.163 HIL: Dialogue box 'Converter Variables'

Note that the ADC full scale voltage has been changed (via a hardware modification) from its original value of 63.2 V to 30.66 V to better utilize the ADC converter for voltage measurement (see equation (4.5)).
A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.161 is the `Main motor control` module,

which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 4.164. Of these, the 'Current controller' has been discussed previously, for a different MCU (see figure 4.124), where it was used in phases : SIM, PIL and COSIM of the development tool-chain.

The per unit variables, DC bus voltage and half-bridge currents are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 40 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `IHM07` with `F303RE` MCU

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which requires that the `Main motor control` module is executed after every fourth (for this drive) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

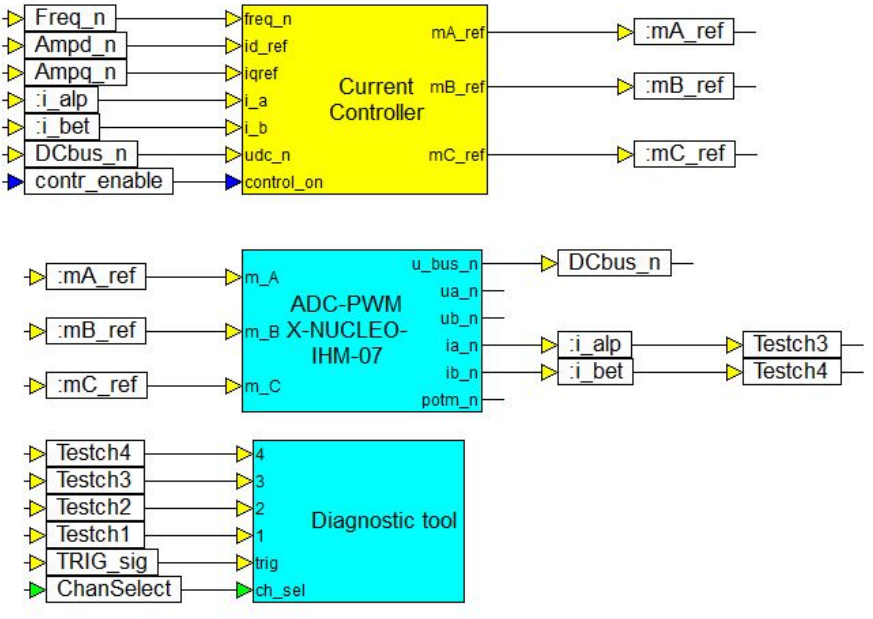


Fig. 4.164 One level down into the 'Main motor controller' module

The `ADC-PWM X-NUCLEO-IHM-07` unit is used to generate the PWM and to

measure the DC bus voltage `DCbus_n`. Furthermore, the three measured phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously.

The HIL-debug/run tool-chain component is the operational component of this example (see figure 4.165), and its task is to execute the `XXX.out` file compiled in the 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct and quadrature current reference values and the shaft speed. Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

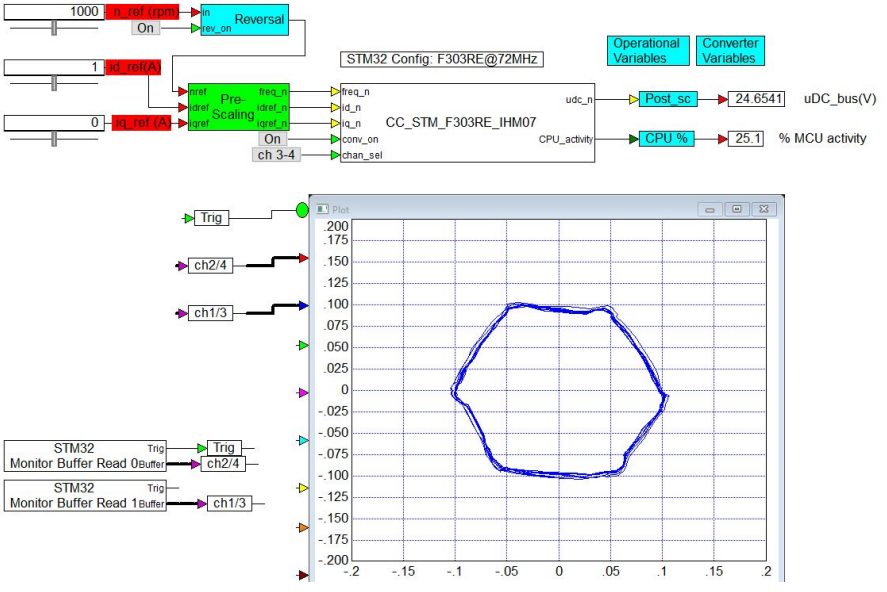


Fig. 4.165 HIL-Debug model, F303-IHM07 of a BLDC drive with current & frequency control

The plot given in figure 4.165 displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit voltage vector locus that is calculated by making use of the three phase modulation indices generated by the current

controller and converting these to stationary variables `m_alp`, `m_beta` using a forward Clarke transformation. These are then subsequently multiplied by the per unit DC voltage divided by two, to arrive at the per unit voltage vector. The converter used in this example does not (by default) use low-pass filters to measure the converter outputs, as is for example the case for the `ESC001` unit (see figure 4.96).
Note that the current reference slider `id_ref` is set to 1.0 A (and `iq_ref`=0), hence the per unit current amplitude should be $|\vec{i}|_{\text{ref}} = {}^{1.0}/_{10} = 0.1$, given that a full-scale current of i_{fs}=10 A is used. Observation of the plot result confirms that the amplitude is approximately equal to the anticipated value. Some amplitude variation will inevitably occur due to the fact that a BLDC machine is attached, which implies that a 6th harmonic component in the PM flux is present. This in turn means that the current locus becomes more hexagonal (compared to circular in a PMsyn), as is indeed the case. Furthermore, the motor coil time constant ${}^{L_s}/_{Rs}$ is relatively low 116 μs, hence the need for a 40 kHz PWM frequency to maintain an acceptable current ripple component.

For comparative purposes an oscilloscope [20] with DC current probe was used to measure the phase 'A' current, as shown in figure 4.166.

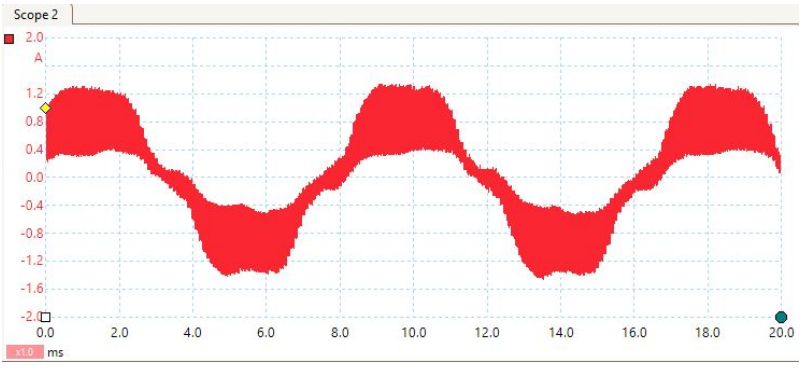


Fig. 4.166 Experimental results: measured phase current, phase 'A'

The measured current amplitude is ≈1.0 A, which corresponds to a per unit value of ${}^{1.0}/_{10} = 0.1$, that is in agreement with the current locus in figure 4.165 (shown on the plot when selecting channel `3-4`). Note also the presence of a current ripple component, as may be observed from figure 4.166.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.2.2.5 Current & Frequency control-F303-ESC001: HIL

Reference files:

- `C&F_F303RE_ESC001_compile.vsm`, for HIL-compile
- `C&F_F303RE_ESC001_run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.167 represents the `ESC001` module, which is connected to a BLDC motor[31]. For HIL operation the drive board itself needs to be known in considerable detail. Aspects need to be considered, as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs, which activate the converter switches. A compile stage, is used to generate the file `XXX.out`, that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for the drive operation with a given controller configuration. The model given in figure 4.168 shows the current controller: `Open loop current Controller` module, which must be compiled to generate an out-file . Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct and quadrature reference currents `idref`, `iqref` and the shaft speed reference `nref`, which are converted to 'per unit' values in the

Fig. 4.167 HIL-setup F303 MCU- ESC001 converter

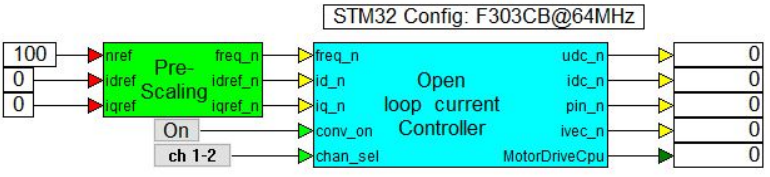


Fig. 4.168 HIL-Compile model of a PMsyn drive with current & frequency control

'pre-scaling' module.
Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Outputs of the Controller module are the per unit DC bus voltage `udc_n`, DC bus current estimate `idc_n`, machine input power `pin_n` and current vector amplitude `ivec_n`, which must be scaled by the 'post-scaling' modules (in the HIL-debug phase) to provide a reading in actual values, e.g Voltage/current and Watts. Furthermore, a `CPU` output is added, which provides the user with a % reading of MCU activity, relative the controller sample time.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes shown in figure 4.169.
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.170), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference currents-frequency.

The 'Converter Variables' dialogue box, as given in figure 4.171, defines the

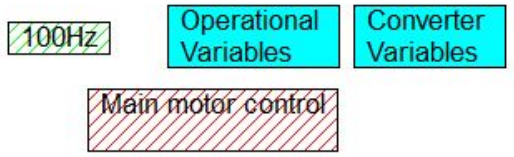


Fig. 4.169 One level down into the Current Controller module

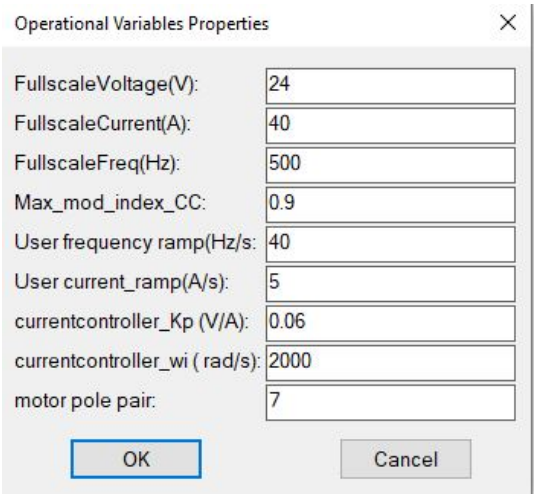


Fig. 4.170 HIL: Dialogue box 'Operational Variables'

scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.4).

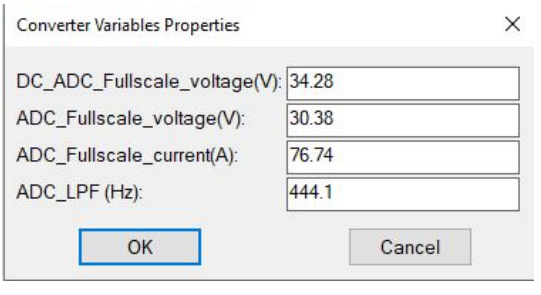


Fig. 4.171 HIL: Dialogue box 'Converter Variables'

Note that the ADC full scale voltage for the DC bus voltage is different to the value used for the converter voltage measurements.
A 'Diagnostic tool' module is used to buffer two variables, which can be

displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.169 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit.
Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 4.172. Of these, the 'Current controller' has been discussed previously, for a different MCU (see figure 4.124), where it was used in sections: SIM, PIL and COSIM of the development tool-chain.

The per unit variables: DC bus voltage and half-bridge currents are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 30 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0.2 μs
- Board in use: `ESC001` with `F303CB` MCU

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which requires that the `Main motor control` module is executed after every third (for this drive) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.
The `ADC-PWM ESC001V1` unit is used to generate the PWM and to measure the DC bus voltage voltage `DCbus_n` as well as the per unit low-pass filtered (and converted via a forward Clarke transformation) converter voltages `u_alp`, `u_bet` . Furthermore, the three measured phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively.

Both measured voltages and currents are used by the input power module `Pow_in` to generate an estimate for the (per unit) input power `pIN_n`. Said variable is also used to generated an estimate for the (per unit) DC bus current `i_DC_n`. A module `amp` is used to calculate the per unit vector current amplitude `IVECA`. ADC Offsets in the ADC unit are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously.

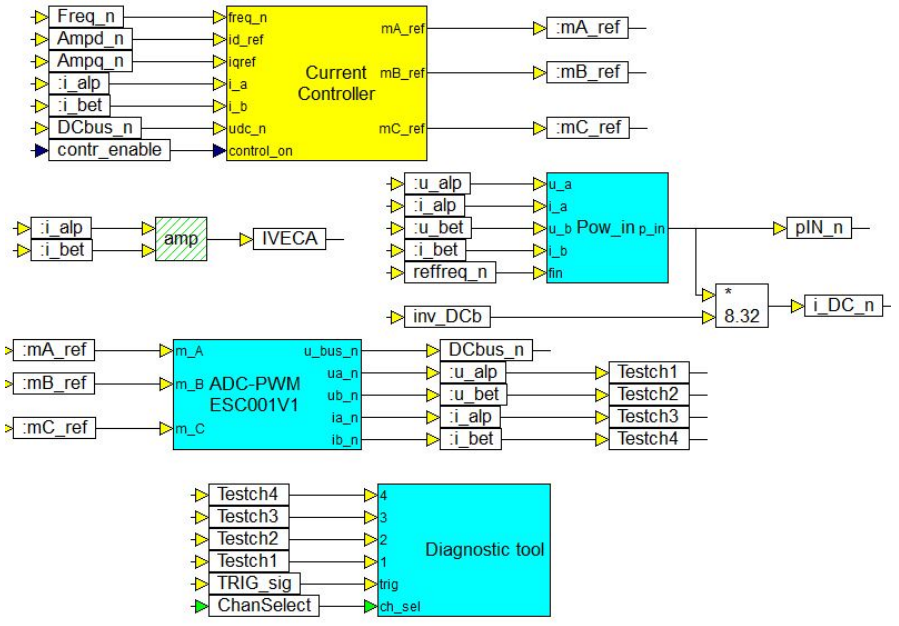


Fig. 4.172 One level down into the 'Main motor controller' module

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 4.173 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).

Central to this figure is the Embed 'target execution module' , which executes said out-file. Three sliders are used to set the direct and quadrature current reference values and the electrical frequency. Post-scaling modules are used to convert the per-unit measured DC bus voltage, DC bus current, current vector amplitude, input power and % CPU time to actual values, as shown with numeric displays.
Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

The plot given in figure 4.173 displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit voltage vector locus that is calculated by making use of low-pass filtered converter voltages and conversion to stationary coordinates using a forward Clarke transformation. Note that the current reference slider `id_ref` is set to 1.0 A (and `iq_ref`=0), hence the per unit current amplitude should be $|\vec{i}|_{\text{ref}} = {}^{1.8}/_{40} = 0.045$, given that a full-scale current of i_{fs}=40 A is used. This value is also shown on the vector

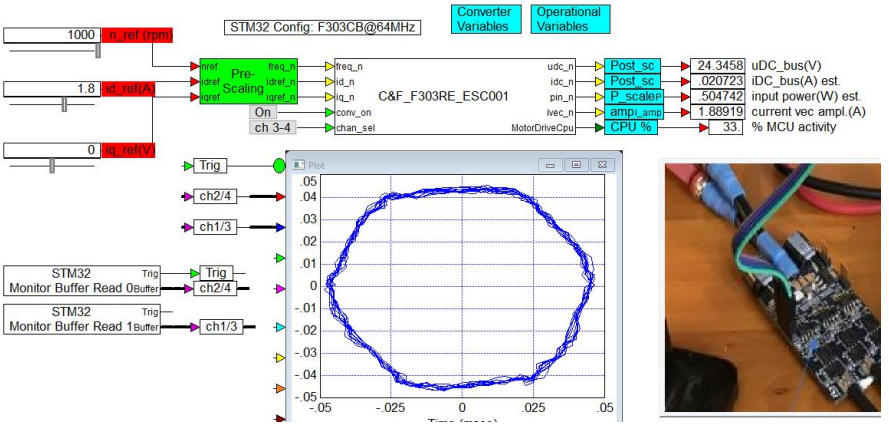


Fig. 4.173 HIL-Debug model, F303CB-ESC001 of a BLDC drive with current & frequency control

amplitude numeric display. Observation of the plot result confirms that the amplitude is approximately equal to the anticipated value. Some amplitude variation will inevitably occur due to the fact that a BLDC machine is attached, which implies that a 6th harmonic component in the PM flux is present. This in turn means that the current locus becomes more hexagonal (compared to circular in a PMsyn) as is indeed the case. Furthermore, the motor coil time constant $^{L_s}/_{Rs}$ is relatively low 116 μs, hence the need for a 30 kHz PWM frequency to maintain an acceptable current ripple component.

For comparative purposes an oscilloscope [20] with DC current probe was used to measure the phase 'A' current, as shown in figure 4.174.

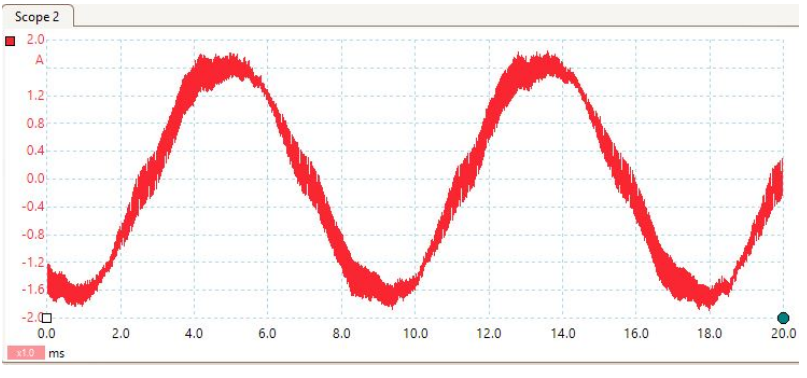


Fig. 4.174 Experimental results: measured phase current, phase 'A'

The measured current amplitude is ≈1.8 A, which corresponds to a per unit value of $^{1.8}/_{40} = 0.045$, that is in agreement with the current locus in figure 4.173 (shown on the plot when selecting channel `3-4`). Note also the presence of a current ripple component, as may be observed from figure 4.174.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.3 Sensored Field Oriented Control with encoder

The term 'field-oriented control' (FOC) implies that the synchronous 'd,q' coordinate system, introduced in previous examples (see sections 4.1, 4.2), is tied to the PM flux vector of the PMsyn or BLDC machine (see section 2.1.3.2). More specifically, the 'd-axis' of said coordinate system is aligned with the flux vector $\vec{\psi}_{\mathrm{PM}}$, as may be observed from figure 4.175. To achieve the best possible torque for the lowest current (referred to as the Maximum Torque Per Ampere (MTPA) operating mode), the current vector should be chosen orthogonal to the d-axis, which therefore implies that the reference current i_d^{ref} is typically set to zero, unless so called 'field-weakening' is used to extend

the operating speed limit of the drive (assuming that the machine in question has the capability to do so, which depends on the ratio of the rated current to the short-circuit current, see equation (2.10)). Furthermore, the direct axis current becomes meaningful, when considering non-salient (IPM) machines, where operating in the MTPA mode requires correct orientation of the current vector in the 'd-q' coordinate frame for a given user torque value, as will be discussed at a later stage.

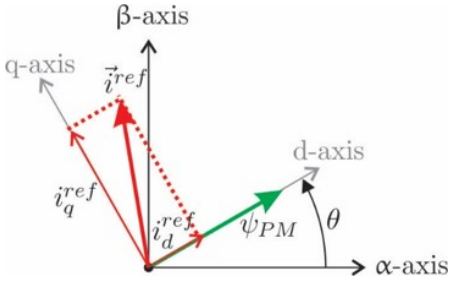


Fig. 4.175 Vector diagram: sensored field-oriented control (FOC)

The term 'sensored' FOC with encoder'(Enc), implies that a shaft encoder is used to provide the angle θ needed to orientate the 'd-axis' relative to the magnetic flux vector of the PM motor. For this type of application, an incremental shaft encoder is used, which typically (for the examples considered below) generates 4000 counts/revolution [2]. In addition, the shaft encoder produces an index pulse (one pulse/per revolution), which provides the ability to use the incremental encoder for absolute angle sensing purposes (this implies that this function becomes active once an index pulse has been identified).

Note also, that in many field-oriented control examples open loop current & frequency control is often used (and added to the examples) as a method to check alignment of the controller 'd-axis' with the PM flux vector, when operating open loop (under no mechanical load conditions). Note that 'Hall-sensors' can also be used for sensored FOC control and this approach is discussed in a separate section (see section 4.4). The tool chain as introduced in figure 2.43 is used as a template for the following subsections.

4.3.1 Sensored Field Oriented Control & encoder-TI

The purpose of this section is to realize field-oriented control with an encoder using a PMsyn or BLDC machine and Texas Instruments hardware. The tool chain as introduced in figure 2.45 is used as a template for the following subsections, hence the toolchain: SIM→PIL→COSIM→HIL, will initially be

considered for a specific hardware example. For this purpose the `F0049-GANINV` module connected to a Teknic [2] PMsyn machine has been chosen. However, additional HIL examples (with different TI hardware) will also be discussed.

4.3.1.1 Sensored FOC with Enc-F0049-GANINV: SIM

Reference file: `FOC_PM_Enc_F0049-GANINV-SIM.vsm`

This example considers the simulation of a PMsyn Teknic motor [2] connected to a Texas Instruments `F0049 GANINV` converter, as shown in figure 4.176. The aim is to examine FOC operation of the drive using a fixed point controller, that will subsequently be used in the HIL implementation.

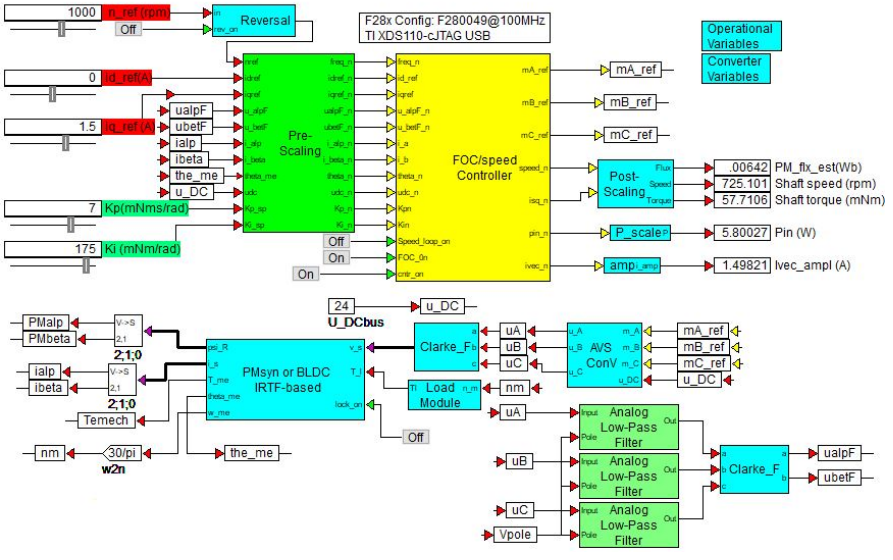


Fig. 4.176 Field-oriented torque/speed control with encoder: simulation

Inputs are the direct and quadrature axis reference currents `id_ref`,`iq_ref` and shaft speed reference `n_ref`. Three logic inputs are shown that provide the following functionality:

- `Ctrl`: converter on/off control.
- `FOC_on`: Field-oriented control active using encoder signal. Operation under 'torque control'.
- `Speed_loop_on`: Field-oriented control active using encoder signal. Operation under 'speed control'.

When the logic inputs `FOC_on` and `Speed_loop_on` are both OFF, drive control is C&F as discussed in section 4.2. Note when operating under torque control, the quadrature axis current reference controls the torque (see equation (2.7a)). If the PM flux vector is fully aligned with the d-axis (as is the case here), changes to the direct axis current reference will not effect the torque. When operating under speed control, the speed controller generates a quadrature axis current reference and adjusts torque levels to meet the criteria: measured shaft speed equal to the user defined reference speed. Also shown in figure 4.176 are the three LPF modules `Analog Low-pass filter` that are used in the GANINV module to remove the PWM component. Output of the forward Clarke converter variables `ualpF`, `ubetF` are used by the controller to calculate motor input power.

Outputs of the controller are three modulation indices, that control the three converter half bridges. The voltages supplied to the motor `uA`, `uB`, `uC` (see figure 4.176) are found by multiplying the modulation indices `mA_ref`, `mB_ref`, `mC_ref` by half the DC bus voltage, as executed in module `AVS ConV`. These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables. The aim of the switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sub chapter 2.1.5). A quadratic load is introduced, which provides a value of 100 mNm at a shaft speed of 1000 rpm.

The IRTF based PM model [13] has a user option to select a synchronous or BLDC type machine, as discussed earlier (see figure 2.12). When selecting the 'BLDC' option a sixth harmonic component is added to the PM flux locus. In this case the 'PMsyn' option is selected, given the machine used for this example.

Drilling into the `FOC/speed Controller` reveals the structure shown in figure 4.177.
Readily apparent is the synchronous current controller module, as discussed earlier, see figure 4.125. Both proportional gain, bandwidth and the maximum modulation index of said controller must be provided in the 'operational variables' dialogue box. The speed controller module, as introduced earlier (see section 3.3), must be provided with appropriate proportional and integral gain values, that depend on the combined load inertia and user defined speed control bandwidth (see equation (2.4)). A set of sliders is introduced in figure 4.176 for this purpose.

The results shown in figure 4.178, represents operation under torque control, hence the steady state shaft speed is dependant on the applied load torque. Note that at start up ramp functions are used, which determine the rate of reference currents-shaft speed changes, as set in the 'Operational Variables' dialogue box.

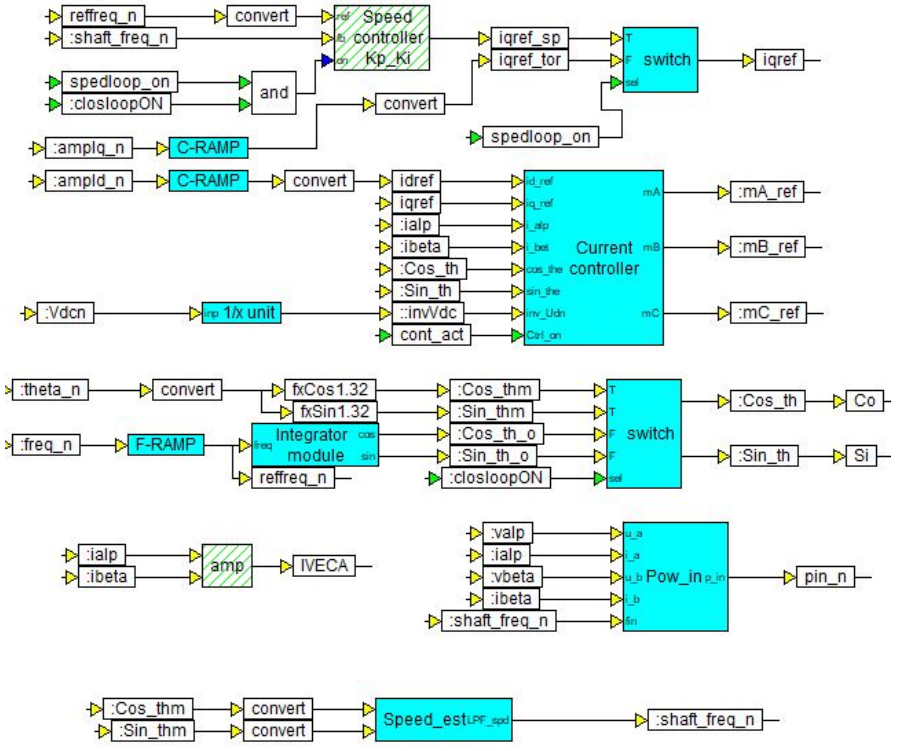


Fig. 4.177 FOC/speed controller structure

An example with speed control active, as given in figure 4.179, shows that the speed controller current limit 2.5 A (set in the 'Operation Variables' dialogue box) is initially reached and around the time mark of 100 ms, the reference speed (set to 800 rpm) is reached. Correspondingly, the speed controller then provides the quadrature current needed to generate a shaft torque that matches the load torque.

Of importance is to consider the vector plot, see figure 4.180 at time mark 250 ms (with the drive operating under speed control), which shows the orientation of the unity d-axis ('green'), voltage vector ('blue'), PM flux vector ('brown', scaled by a factor 500) and the current vector ('red'). Observation of the flux vector confirms that it is indeed aligned with the controller d-axis and the current vector is aligned with the q-axis, as expected. Note also, that the voltage vector leads the current vector, which is expected due to the inductive nature of the machine. Note that the phase lead of said voltage vector is determined by the voltage across the inductance and this is limited for the machine in question because the short-circuit current is high in comparison to the current amplitude used (see equation (2.10)).

Operation of the drive is at 800 rpm and the load controls the quadrature

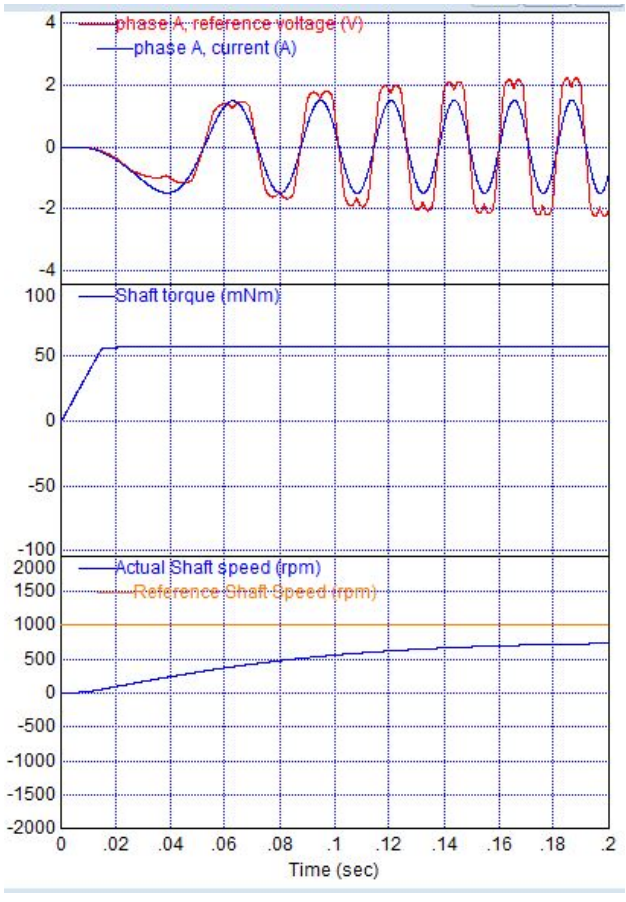


Fig. 4.178 SIM result, PMsyn operation, FOC-torque control active

current amplitude, as mentioned earlier (because speed control is active).

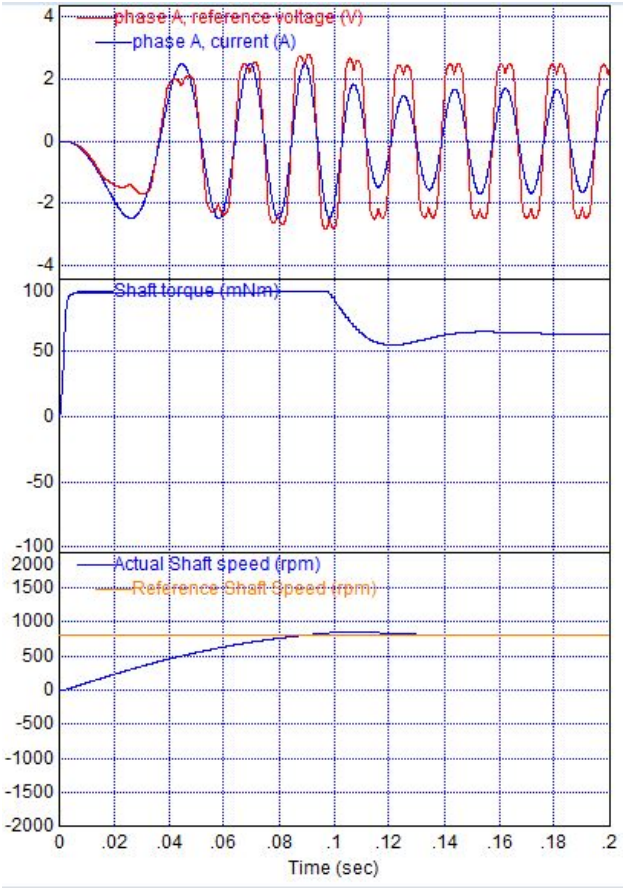


Fig. 4.179 SIM result, PM syn operation, FOC-speed control active

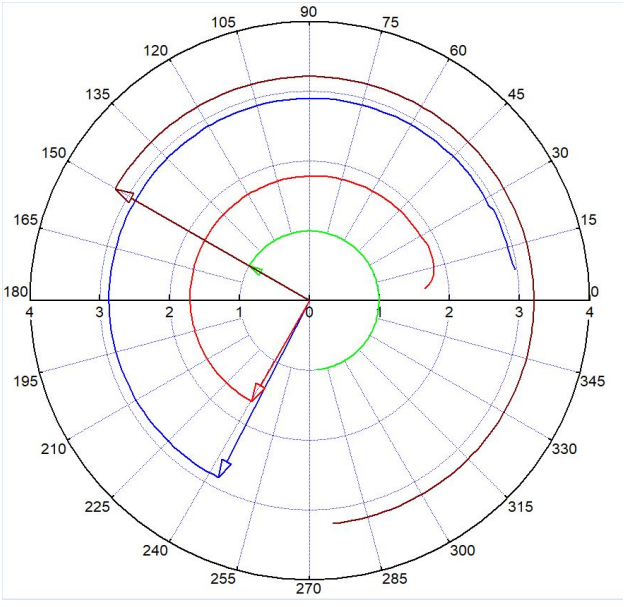


Fig. 4.180 SIM result, PM syn operation, FOC-speed control active: vector plot

4.3.1.2 Sensored FOC with Enc-F0049-GANINV: PIL

Reference files:

- `FOC_PM_Enc_F0049-GANINV-PIL.vsm`
- `FOC_enc_PIL_mod.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F0049. This implies that the controller module shown in the simulation (see figure 4.176) is now replaced with a target execution module for file: `FOC_enc_PIL_mod.out` as indicated in figure 4.181.

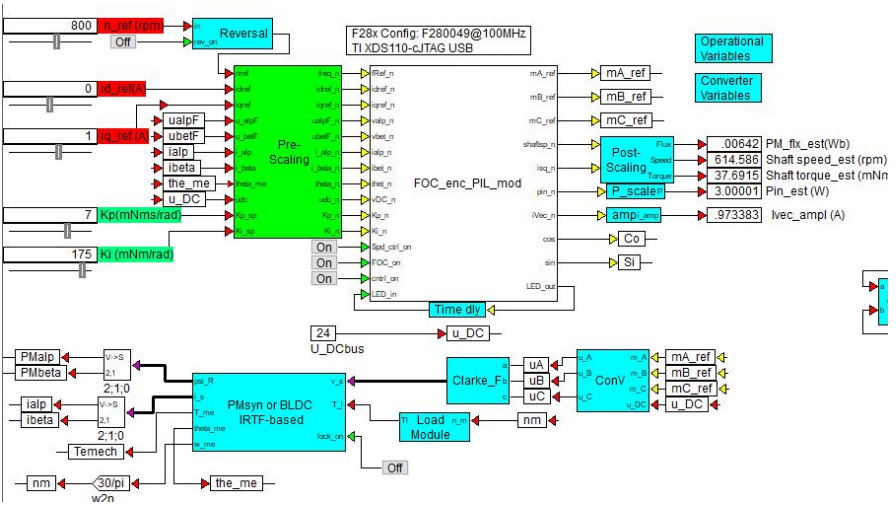


Fig. 4.181 Field-oriented torque/speed control with encoder: PIL operation

The target interface model that executes the out-file, generates the modulation indices required for the converter motor and these components are not affected by use of the PIL. Generation of the out file is based on the PIL module (see figure 4.182), which must be compiled using the command: `toolbox>codegen`.

Moving one level into the PIL module `FOC_enc_PIL_mod` (see figure 4.183), shows the FOC/speed controller as introduced in section 4.3.1.1. Added to this PIL unit is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that PIL control is operational. In addition, two variables `cos` and `sin`, are provided as outputs. These are used in a vector plot to show the 'd' axis of the controller algorithm.

Running the PIL file by starting the file ('green arrow') will show an active

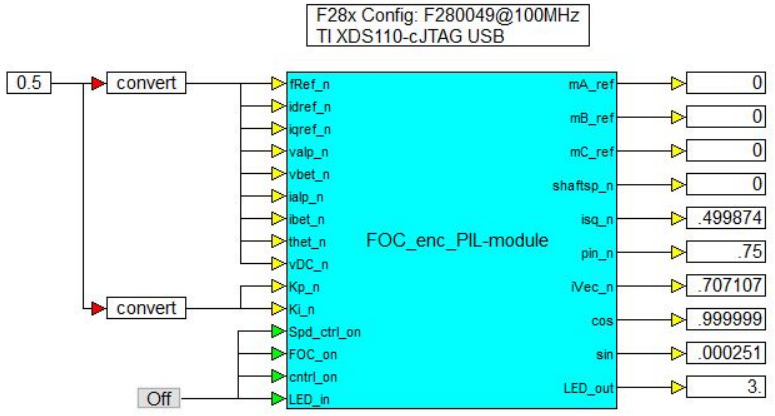


Fig. 4.182 PIL module, current & frequency control

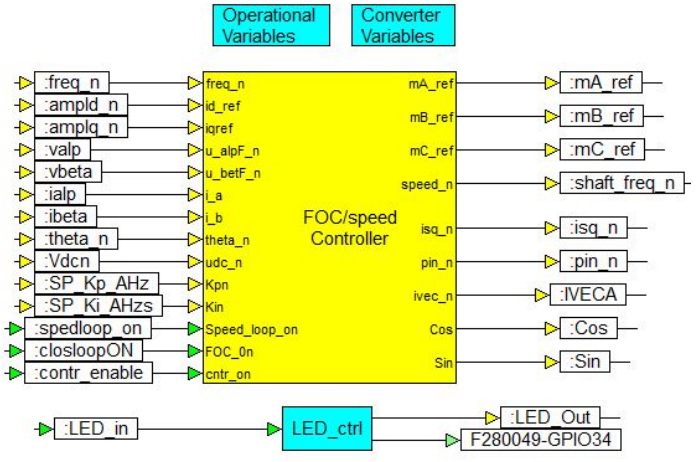


Fig. 4.183 Content PIL module, Field-oriented torque/speed control with encoder

'red' LED on the Launchpad, which indicates that the MCU is operational. Results produced by PIL operation (running in this case with FOC-speed control active) are identical to the plots shown in figures 4.179, 4.180 and are therefore not shown here.

4.3.1.3 Sensored FOC with Enc-F0049-GANINV:COSIM

Reference files:

- `FOC-enc_F0049_GANINV-COSIM.vsm`
- `GAN_INVv12.psimsch`
- `my motor model_v3.psimsch`

To evaluate operation of a controller with a switching converter, in this case the `GANINV`, use is made of a `PSIMCoupler` module, as indicated in figure 4.184. This approach implies the use of PSIM together with Embed, which is referred to as 'co-simulation'.

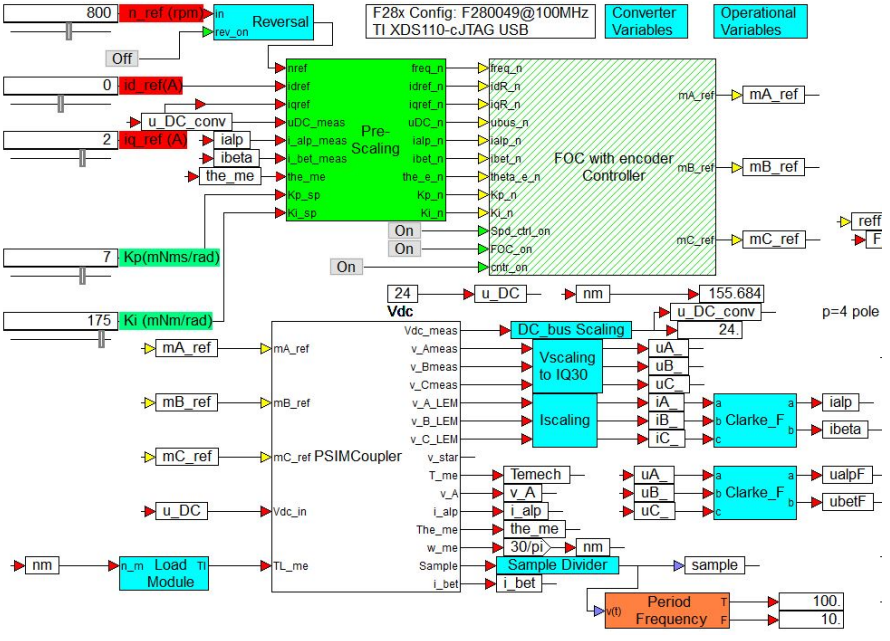


Fig. 4.184 COSIM based drive, field-oriented torque/speed control with encoder

The `FOC with encoder controller`, hatched in this case, as its sampling frequency is locally set to 10 kHz, whereas the 'system' frequency (set in the 'system properties' dialogue box) is set to 20 MHz. The latter is much higher, given the need to accurately represent 20 kHz voltage switching waveforms generated by the PWM unit of the converter. The content of the `FOC with encoder controller` module was introduced earlier (see figure 4.177). However, the current and voltage measurements now make use of the signal processing circuitry connected to the half-bridges.

In this case the current transducers are in the converter phases, as shown earlier (see figure 4.10). After sampling and signal-processing in the converter, these variables appear as: `iA_`, `iB_`, `iC_` in figure 4.184. The low pass filtered voltages (which removes the PWM component) of the three half-bridges is also present in figure 4.184 and are scaled and sampled by the module `Vscaling to IQ30`. In this case the per unit variables are re-scaled to actual values: `uA_`, `uB_` and `uC_` respectively.
The 'Converter Variables' dialogue box (shown in figure 4.184), defines the scaling used by the Analog to Digital converter, which has a maximum input value of 3.3 V and is directly linked to the signal processing undertaken for voltages and currents as discussed above. For example, the 'ADC-fullscale-voltage' value is the theoretical peak-to peak input voltage which must be applied to the low-pass filter network (see figure 4.12) to ensure that the ADC input level varies between $0 \rightarrow 3.3$V. The current transducer voltages `v_A_LEM`, `v_B_LEM` and `v_C_LEM` are sampled by the `Iscaling` module shown in figure 4.184. When dealing with the actual MCU, the ADC converter also takes over this sampling process.
Apart from the measured voltage/currents generated by the converter, the PSIM coupler unit also provides the actual motor model current vector: `i_alp`, `i_bet` (in stationary coordinates), phase 'A' voltage and speed, which are presented by the variables `v_A`, `i_alp`,`i_bet` and `n_m` (after conversion to rpm) respectively, see figure 4.184.

Results achieved with the COSIM approach are shown in figure 4.185.
Indicated in said figure is the phase 'A' voltage waveform, together with the reference voltage, phase current, shaft torque and shaft speed, with the drive operating under speed control.
A vector plot, given in figure 4.186, at time mark 40 ms after starting the co-simulation shows the measured LPF voltage vector ('blue'), unity d-axis vector ('green') and the actual (from the IRTF motor model) current vector ('red'). Note that the impact of switching is clearly reflected in the current vector locus, e.g. changes in end-point direction due to use of 'active' or 'zero' voltage vectors [13].
Co-simulation, as discussed in this sub-section, is instructive for understanding the impact of using a switching converter on drive operation, e.g to asses the impact of shaft torque ripple due to PWM.

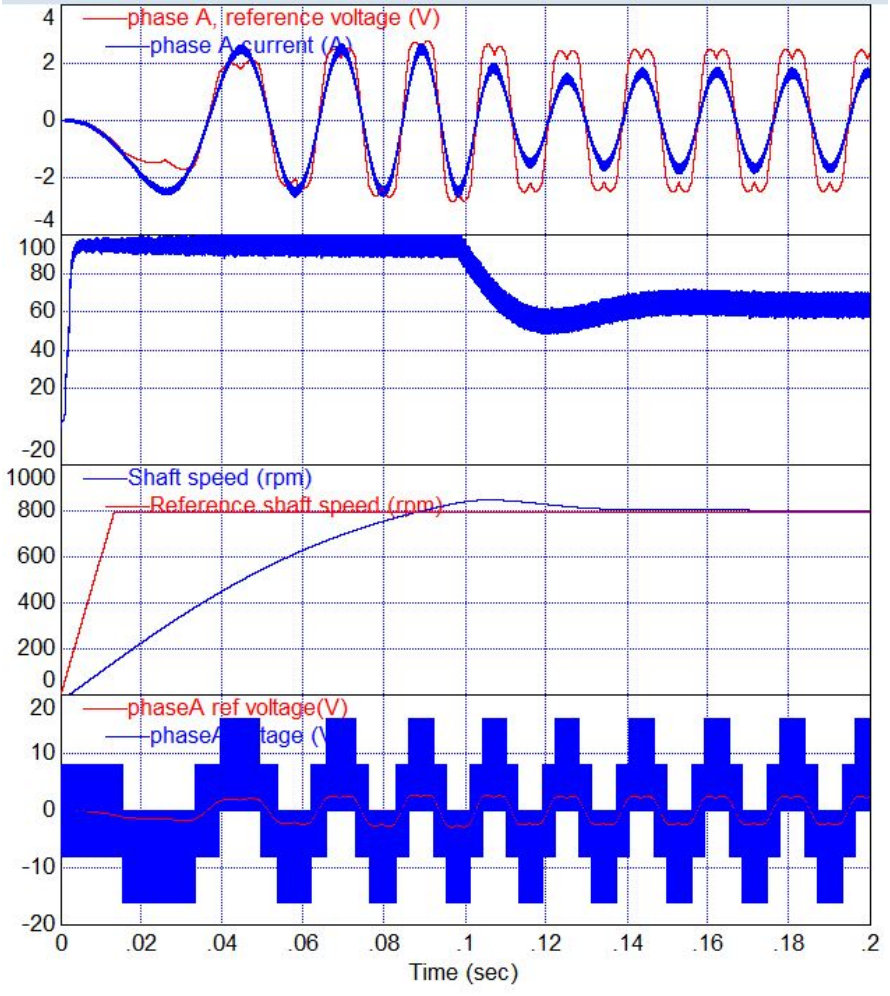


Fig. 4.185 Result of COSIM, field-oriented speed control with encoder

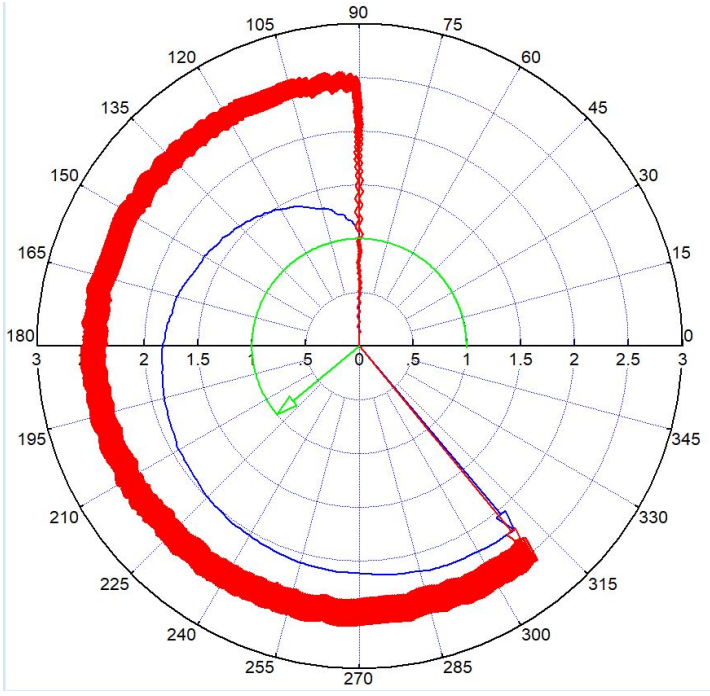


Fig. 4.186 Result of COSIM, field-oriented speed control with encoder: vector plot

4.3.1.4 Sensored FOC with Enc-F0049-GANINV: HIL

Reference files:

- `FOC_sensored_Enc_F0049-GANINV-compile.vsm`, for HIL-compile
- `FOC_sensored_Enc_F0049-GANINV-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.187 represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F280049` module), which is connected to a PMsyn Teknic motor[2].

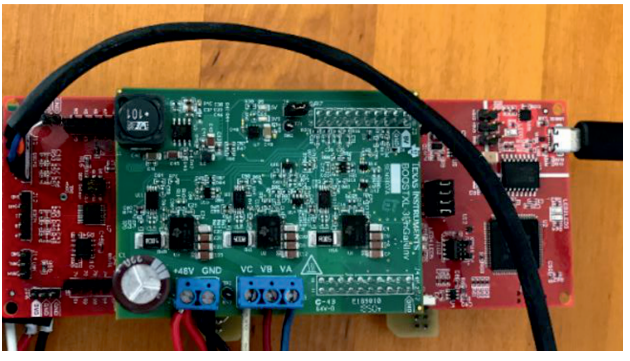


Fig. 4.187 F280049 MCU-GANINV setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.188 shows the 'FOC sensored and C&F control' module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct axis reference current `id`, quadrature axis reference current `iq`, and the reference shaft frequency `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller.
Four logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other two are used to select field-oriented control (FOC) operation `FOC_on` and speed operation under FOC `Spd_ctrl_on`, as discussed

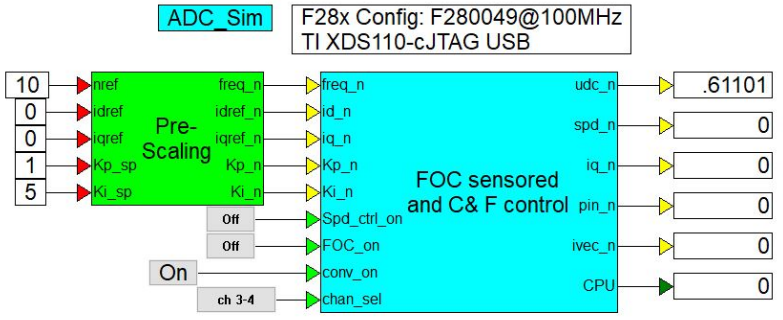


Fig. 4.188 HIL-Compile model of a PMsyn drive with field-oriented control and encoder

earlier (see section 4.3.1.1).

Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- A per unit shaft speed estimate `spd_n` generated via the encoder.
- The measured per unit quadrature current `iq_n`, used to generate a shaft torque estimate.
- A per unit estimate of the motor input power `pin_n`. This is also useful as an estimate for the drive input power (bus this ignored the converter losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx p_{\mathrm{in}}/u_{\mathrm{DC}}$.
- The measured per unit current vector amplitude `ivec_n`.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 4.188 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Hence these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the controller module ('right' mouse click, after selecting module), leads to the set of modules and dialogue boxes shown in figure 4.189.
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.190), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage-frequency. In addition, the speed controller quadrature current (absolute value) has been

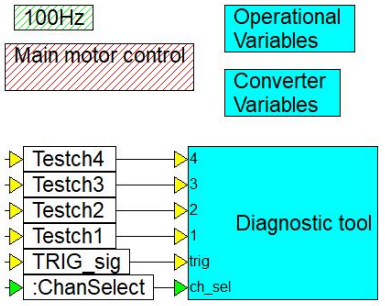


Fig. 4.189 One level down into the FOC sensored Controller module

added, which in effect sets the shaft torque limit for the drive. Furthermore, the speed controller sampling frequency is typically set lower then used by the main controller. In this case it is set to 1 kHz. Also added to this dialogue module are motor parameters: motor pole pair number and PM flux (as for example found earlier, see the 'Post Analysis' 4.2.1.5).

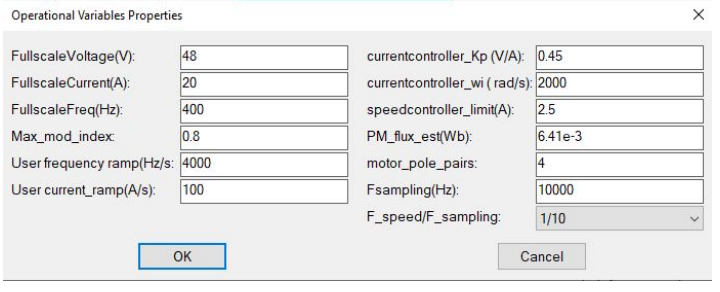


Fig. 4.190 HIL: Dialogue box 'Operational Variables'

The 'Converter Variables' dialogue box, as given in figure 4.191, defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.2).
This dialogue box also shows the low-pass filter corner frequency used by the converter for measuring/ LPF filtering the half-bridge voltages (see figure 4.191).
A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot, when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

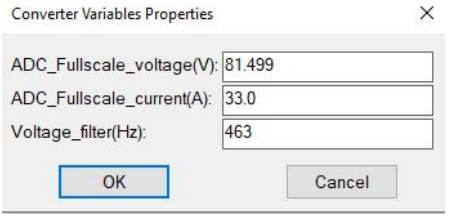


Fig. 4.191 HIL: Dialogue box 'Converter Variables'

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.189 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by every second (for this drive) 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 4.192. Of these, the 'FOC/speed controller' has been discussed earlier (see figure 4.177), where it was used in phases: SIM, PIL and COSIM of the development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F280049` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present, which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after every second (for this drive) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.
The `ADC-PWM-LaunchXL-28004x BOOSTXL-3phGANINV` unit is used generate the PWM signals for the converter. In addition it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively.

ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box

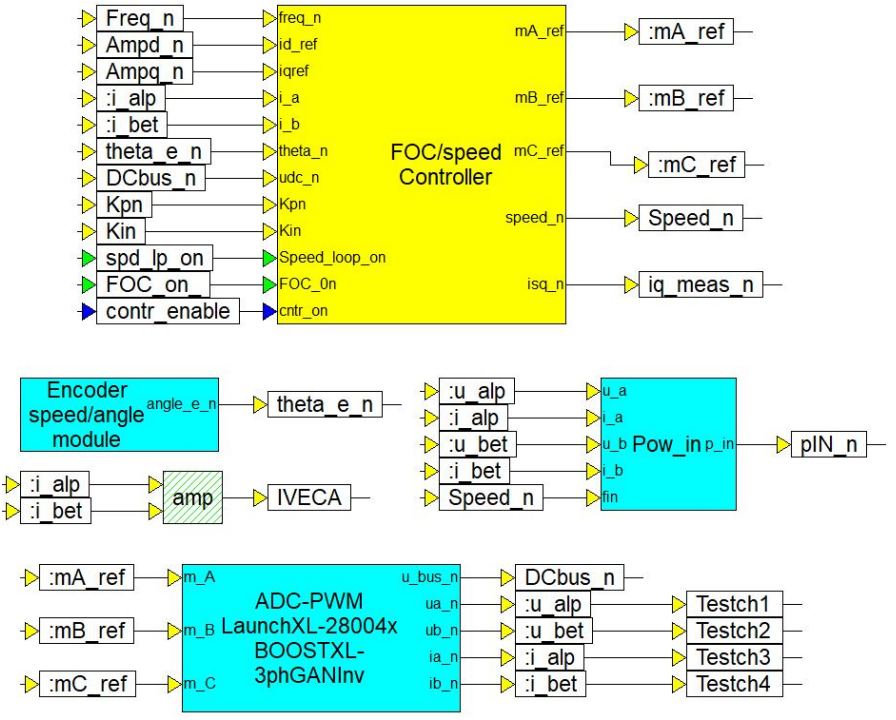


Fig. 4.192 One level down into the 'Main motor controller' module

of the ADC/PWM module). Estimation of the ADC offsets is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage `DCbus_n_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 4.193 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct and quadrature axis current reference values and the electrical frequency. In addition, two sliders are present to allow the user to set the proportional and integral gain of the speed controller.
Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Furthermore, numerical displays are present to show: PMflux, estimated shaft speed, shaft torque, input power and current vector amplitude. To monitor drive operation a set of gauges have been added, which show (low pass filtered): shaft torque, input power, shaft speed and an estimate for the DC supply current.

Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

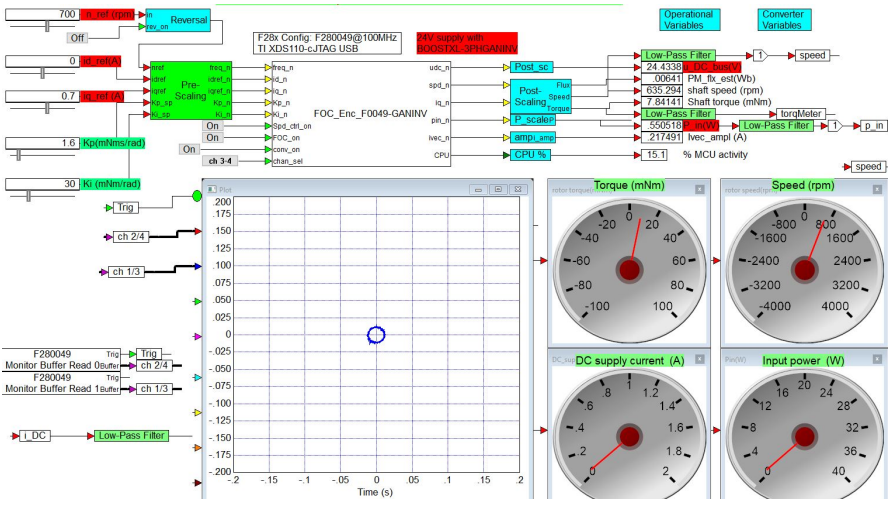


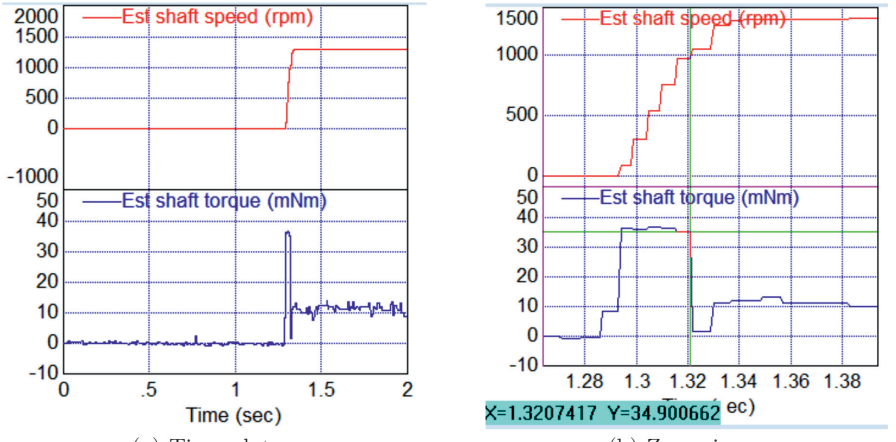
Fig. 4.193 HIL-Debug model, F0049-GANINV of a PMsyn drive with field-oriented control using an encoder

The plot given in figure 4.193 displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit LPF voltage vector locus. The electrical frequency reference in figure 4.193 is set to 700 rpm. These results shows operation under FOC-speed control with the machine operating under no-load conditions. An instructive exersize for the reader is to switch operation to FOC 'torque control' with both direct and quadrature current reference initially set to zero. Then hold the rotor by 'hand' and feel the effect of torque application when moving the quadrature reference slider. Then repeat this exersize by moving the direct axis reference (with the quadrature slider at zero), in which case NO torque on the shaft should be present (assuming that the PM flux is aligned the d-axis of the controller).

Post Analysis

The issue of getting access to an estimate of the motor (with attached load) inertia is important, given the need to provide said parameter in the SIM, PIL and COSIM examples discussed earlier. Furthermore, the inertia value is also a good guide for setting the speed controller gains, as discussed earlier (see equation 2.4). The test sequence shown here for determining an estimate for the inertia is also helpful for a spot check on the rotor flux parameter in use. Both these issues can be addressed by setting the FOC drive in 'torque mode'

and reducing the maximum modulation index to, for example 0.3, which implies a maximum terminal voltage amplitude of 3.6 V. This is required, because the HIL is run with the quadrature slider set to 1 A and run time of 2 s and these setting limit the steady-state shaft speed. Consequently, when the file is started ('green arrow') AND the button `conv_on` is ON (BEFORE you hit the green arrow) then the machine should initially generate a torque that corresponds to the quadrature slider setting.
A plot (see figure 4.194(a)) showing the shaft speed and torque gives important information.



(a) Time plot (b) Zoom in

Fig. 4.194 HIL-Debug model, F0049-GANINV of a PMsyn drive with torque control active, used for Inertia and flux estimation

At time mark ≈1.28 s the ADC offsets have been found in the ADC module and drive operation commences. A torque step that corresponds to the quadrature slider setting follows (as anticipated) and the machine (in this case no-load) accelerates until the EMF reaches the terminal voltage (ignoring the phase resistance voltage drop) amplitude of 3.6 V. Consequently, the quadrature current controller will move to voltage control and the current decays to a level which generates a shaft torque that matches the ≈10 mNm friction torque. The steady-state shaft speed at time mark 2 s can be found using the 'Read Coordinates' feature of the plot and is equal to 1297 rpm. An estimate for the PM flux can then be found using $\psi \approx {}^{e}/_{(p\,\omega_m)}$, where ω_m is the shaft speed rad/s, which yields a value of $\psi \approx 6.6$ mWb. Note that earlier (see section 4.2.1.5, shown for a different machine to the one used here) a C&F HIL approach was introduced to estimate the PM flux.

The issue of estimating inertia for the machine used in figure 4.194(a) is facilitated by the 'zoom in' on the time slot, where current control is active

(see figure 4.194(b)), i.e. where the shaft torque pulse is present. Observation of this result shows that the torque pulse is active for $\Delta t \approx 27$ ms and during that time a linear speed increase of 1000 rpm takes place with a torque change equal to the shaft torque minus the friction torque, hence $\Delta T \approx 25$ mNm in this case. An estimate for the inertia can thus be found using expression (4.17).

$$J \approx \frac{(T_{\text{me}} - T_{\text{friction}})\ \Delta t}{\Delta \omega_m} \tag{4.17}$$

For the machine used in this example an
inertia estimate of $J \approx 6.4\,10^{-6}\,\text{Kg}\,\text{m}^2$ was found.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.3.2 Sensored Field Oriented Control & encoder-STM

The purpose of this section is to realize field-oriented control with an encoder using a PMsyn or BLDC machine and ST Micro hardware. The tool chain as introduced in figure 2.45 is used as a template for the following subsections, hence the toolchain: SIM→PIL→HIL, will initially be considered for a specific hardware example. For this purpose the `G473E-BDRIVE` module [34][35] connected to a Teknic [2] PMsyn machine has been chosen.

4.3.2.1 Sensored FOC with Enc-G473E-B-DRIVE: SIM

Reference file: `FOC_PM_Enc_G473-Bdrive-SIM`

This example considers the simulation of a PMsyn Teknic motor [2] connected to a ST Micro `G473 Bdrive`, as shown in figure 4.195. The aim is to examine FOC operation of the drive using a fixed point controller, that will subsequently be used in the HIL implementation.

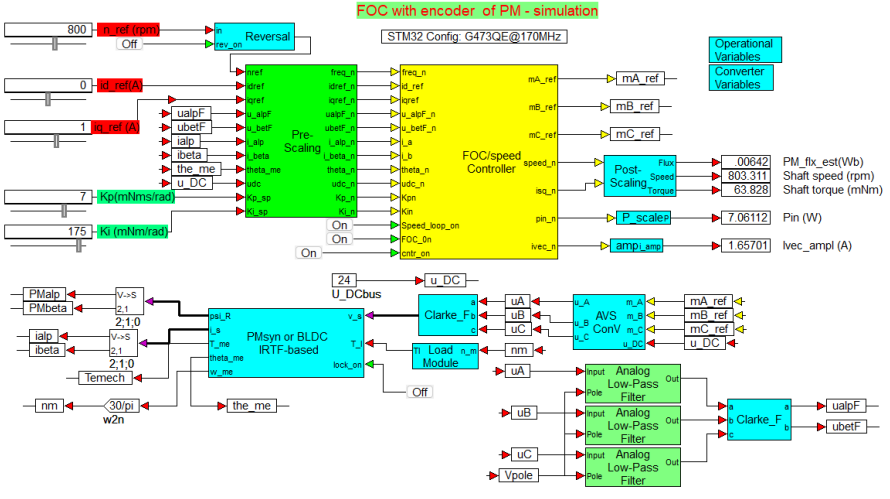


Fig. 4.195 Field-oriented torque/speed control with encoder: simulation

Inputs are the direct and quadrature axis reference currents `id_ref`, `iq_ref` and shaft speed reference `n_ref`. Three logic inputs are shown that provide the following functionality:

- `Ctrl_on`: converter on/off control.

- `FOC_on`: Field-oriented control active using encoder signal. Operation under 'torque control'.
- `Speed_loop_on`: Field-oriented control active using encoder signal. Operation under 'speed control'.

When the logic inputs `FOC_on` and `Speed_loop_on` are both OFF, drive control is C&F, as discussed in section 4.2. Note when operating under torque control, the quadrature axis current reference controls the torque (see equation (2.7a)). If the PM flux vector is fully aligned with the d-axis (as is the case here), changes to the direct axis current reference will not effect the torque. When operating under speed control, the speed controller generates a quadrature axis current reference and adjusts torque levels to meet the criteria: measured shaft speed equal to the user defined reference speed. Also shown in figure 4.195 are the three LPF modules `Analog Low-pass filter` that are used in the B-Drive to remove the PWM component. Outputs of the forward Clarke converter variables `ualpF`, `ubetF` are used by the controller to calculate motor input power.

Outputs of the controller are three modulation indices, that control the three converter half bridges. The voltages supplied to the motor `uA`, `uB`, `uC` (see figure 4.195) are found by multiplying the modulation indices `mA_ref`, `mB_ref`, `mC_ref` by half the DC bus voltage, as executed in module `AVS ConV`. These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables. The aim of the switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sub chapter 2.1.5). A quadratic load is introduced, which provides a value of 100 mNm at a shaft speed of 1000 rpm.

The IRTF based PM model [13] has a user option to select a synchronous or BLDC type machine, as discussed earlier (see figure 2.12). When selecting the 'BLDC' option a sixth harmonic component is added to the PM flux locus. In this case the 'PMsyn' option is selected, given the machine used for this example.

Drilling into the `FOC/speed Controller` reveals the structure shown in figure 4.196.
Readily apparent is the synchronous current controller module, as discussed earlier, see figure 4.125. Both proportional gain, bandwidth and the maximum modulation index of said controller must be provided in the 'operational variables' dialogue box. The speed controller module, as introduced earlier (see section 3.3), must be provided with appropriate proportional and integral gain values, that depend on the combined load inertia and user defined speed control bandwidth (see equation (2.4)). A set of sliders is introduced in figure 4.195 for this purpose.

The results shown in figure 4.197, represents operation under torque control,

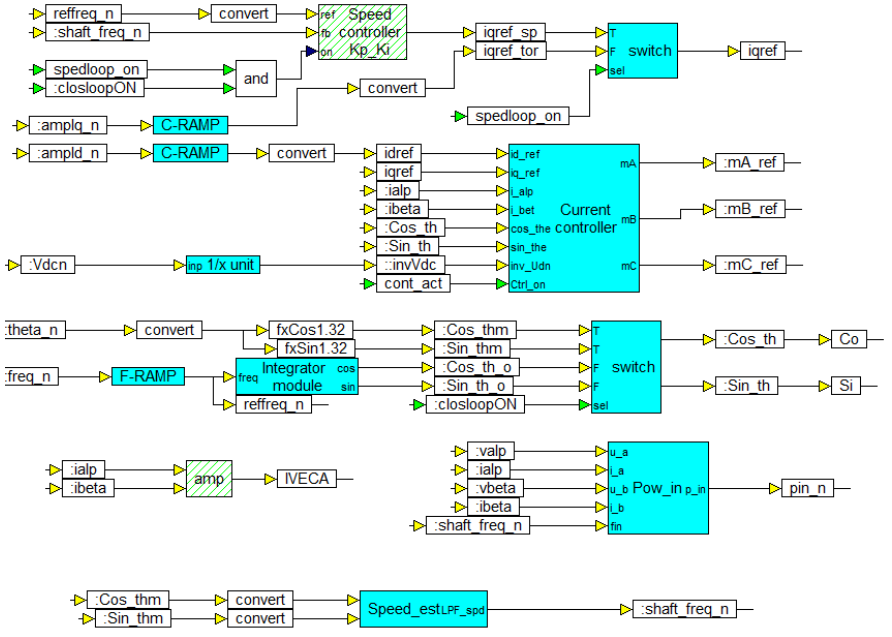


Fig. 4.196 FOC/speed controller structure

with i_q^{ref} =1.0 A, hence the steady state shaft speed is dependent on the applied load torque. Note that at start up ramp functions are used, which determine the rate of reference currents-shaft speed changes, as set in the 'Operational Variables' dialogue box.

An example with speed control active, as given in figure 4.198, shows that the speed controller current limit 2.5 A (set in the 'Operation Variables' dialogue box) is initially reached and around the time mark of 100 ms, the reference speed (set to 800 rpm) is reached. Correspondingly, the speed controller then provides the quadrature current needed to generate a shaft torque that matches the load torque.

Of importance is to consider the vector plot, see figure 4.199 at time mark 200 ms (with the drive operating under speed control), which shows the orientation of the unity d-axis ('green'), voltage vector ('blue'), PM flux vector ('brown', scaled by a factor 500) and the current vector ('red'). Observation of the flux vector confirms that it is indeed aligned with the controller d-axis and the current vector is aligned with the q-axis, as expected. Note also, that the voltage vector leads the current vector, which is expected due to the inductive nature of the machine. Note that the phase lead of said voltage vector is determined by the voltage across the inductance and this is limited for the machine in question because the short-circuit current is high in comparison to the current amplitude used (see equation (2.10)).

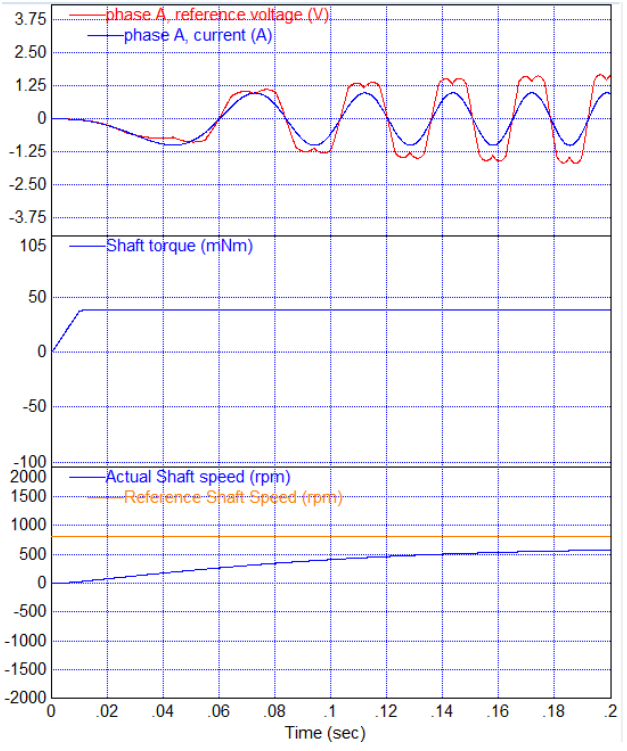


Fig. 4.197 SIM result, PMsyn operation, FOC-torque control active

Operation of the drive is at 800 rpm and the load controls the quadrature current amplitude, as mentioned earlier (because speed control is active).

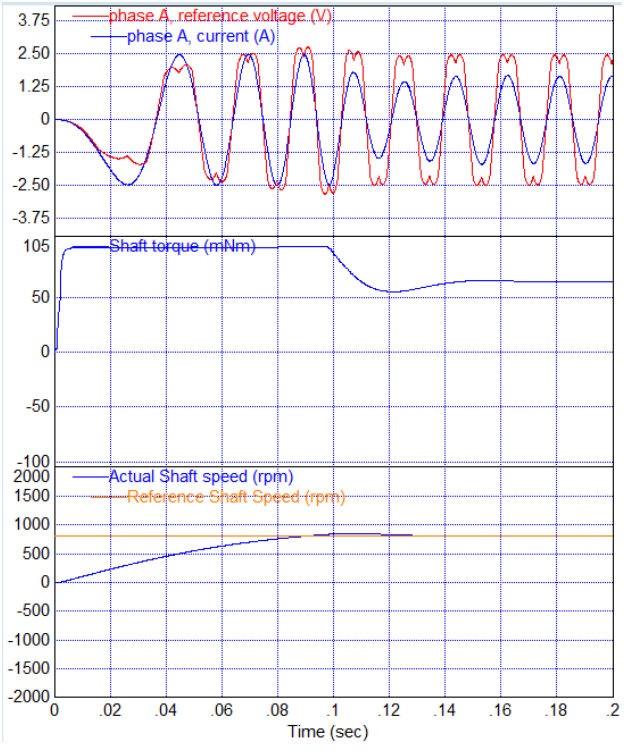


Fig. 4.198 SIM result, PM syn operation, FOC-speed control active

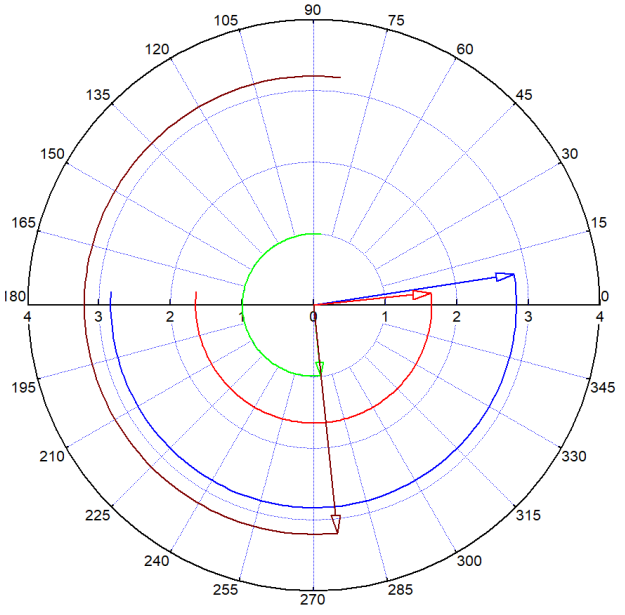


Fig. 4.199 SIM result, PM syn operation, FOC-speed control active: vector plot

4.3.2.2 Sensored FOC with Enc-G473E-B-DRIVE: HIL

Reference files:

- `FOC_sensored_G4_Bdrive-compilev6`, for HIL-compile
- `FOC_sensored_G4_Bdrive-debugv6`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.200 represents the `B-Drive` converter (connected to the `ST Micro-G473` module), which is connected to a PMsyn Teknic motor[2].

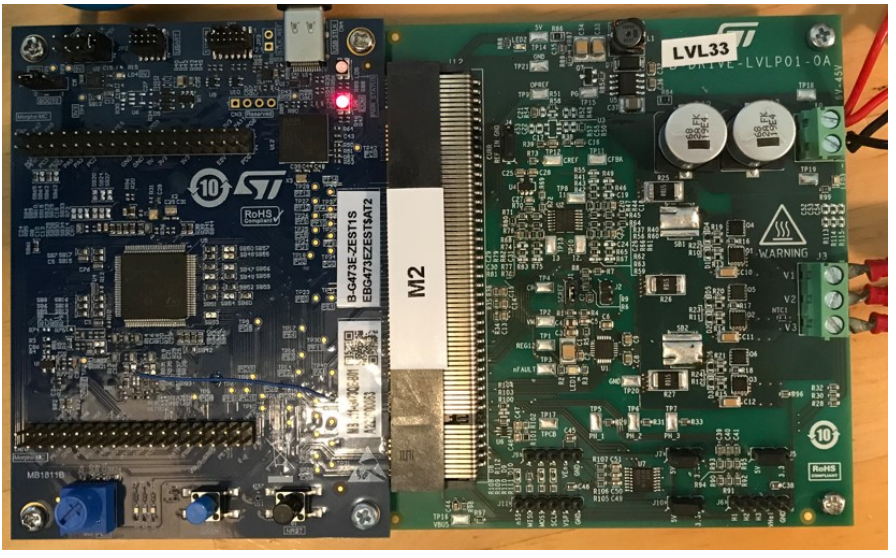


Fig. 4.200 G473E-B-Drive setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.201 shows the 'FOC sensored and C&F control' module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct axis reference current `id`, quadrature axis reference current `iq`, and the reference shaft frequency `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to

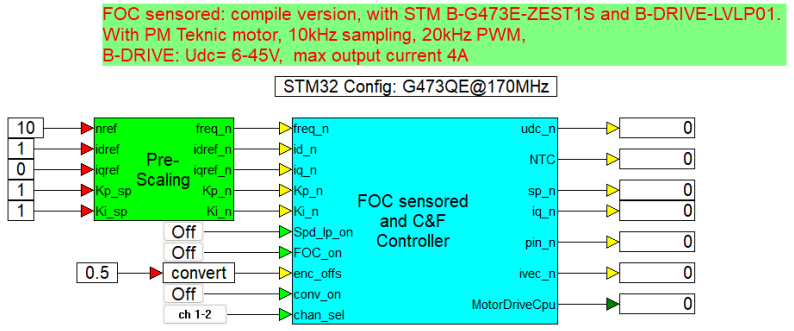


Fig. 4.201 HIL-Compile model of a PMsyn drive with field-oriented control and encoder

tune the speed controller. Furthermore, an input `enc_offs` is provided to setup the encoder (eliminate offsets) using an external slider.
Four logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other two are used to select field-oriented control (FOC) operation `FOC_on` and speed operation under FOC `Spd_ctrl_on`.
Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- A NTC output, used for measuring temperature, power board.
- A per unit shaft speed estimate `spd_n` generated via the encoder.
- The measured per unit quadrature current `iq_n`, used to generate a shaft torque estimate.
- A per unit estimate of the motor input power `pin_n`. This is also useful as an estimate for the drive input power (bus this ignored the converter losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx {}^{p_{\mathrm{in}}}/_{u_{\mathrm{DC}}}$.
- The measured per unit current vector amplitude `ivec_n`.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Moving one level down into the controller module ('right' mouse click, after selecting module), leads to the set of modules and dialogue boxes shown in figure 4.202.
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.203), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage-frequency. In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore,

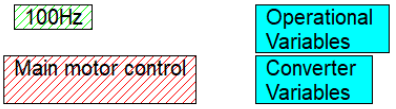


Fig. 4.202 One level down into the FOC sensored Controller module

the speed controller sampling frequency is typically set lower then used by the main controller. In this case it is set to 1 kHz. Also added to this dialogue module are motor parameters: motor pole pair number and PM flux (as for example found earlier, see the 'Post Analysis' 4.2.1.5).

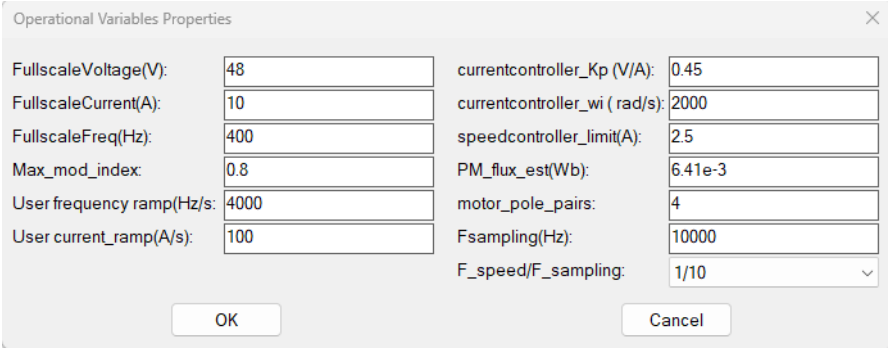


Fig. 4.203 HIL: Dialogue box 'Operational Variables'

The 'Converter Variables' dialogue box, as given in figure 4.204, defines the scaling used by the Analogue to Digital converter, as discussed earlier, see equation (4.2).
This dialogue box also shows the low-pass filter corner frequency used by the converter for measuring/ LPF filtering the half-bridge voltages.

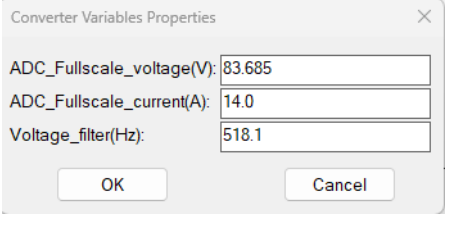


Fig. 4.204 HIL: Dialogue box 'Converter Variables'

A '100Hz' module undertakes all the background tasks for the drive. Among

these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.202 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by every second (for this drive) 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 4.205. Of these, the 'FOC/speed controller' has been discussed earlier (see figure 4.196), where it was used in phase: SIM, of the development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0.2 μs
- Board in use: `G473QE` with `B-Drive` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after every second (for this drive) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

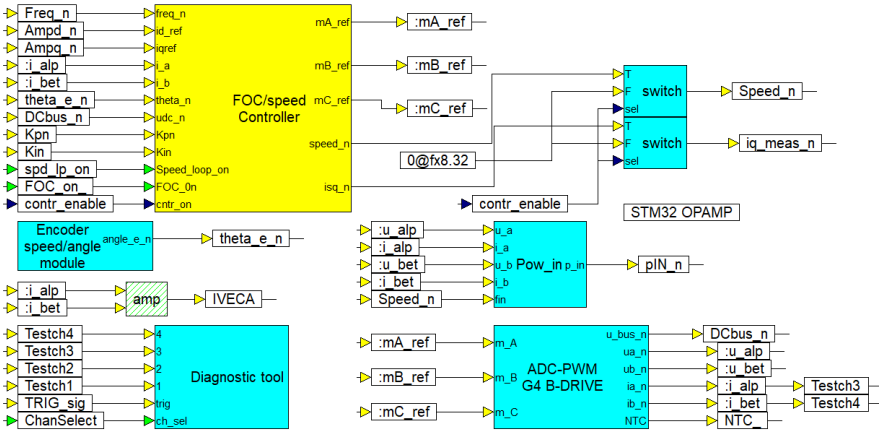


Fig. 4.205 One level down into the 'Main motor controller' module

The `ADC-PWM-G4 B-DRIVE` unit is used generate the PWM signals for the converter. In addition it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates

α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively. In addition, an output variable `NTC_` has been added to allow temperature measurement of the converter.

ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offsets is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage `DCbus_n`, completes the ADC measurement sequence.
A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot, when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 4.206 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct and quadrature axis current reference values and the electrical frequency. In addition, three sliders are present to allow the user to set the proportional/integral gain of the speed controller and manipulate the encoder offset.
Post-scaling modules are used to convert the per-unit measured DC bus voltage, converter temperature and % CPU time to actual values, as shown with numeric displays. Furthermore, numerical displays are present to show: PMflux, estimated shaft speed, shaft torque, input power and current vector amplitude. To monitor drive operation a set of gauges have been added, which show (low pass filtered): shaft torque, input power, shaft speed and an estimate for the DC supply current.

Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.
The plot given in figure 4.206 displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the Cosine of the open loop control angle and the Cosine of the encoder angle (display to be set in time mode). When operating under open loop current and frequency control (with a given direct axis current and zero quadrature current) the two Cosine signals must be aligned. The shaft speed reference in figure 4.206 is set to 800 rpm, and the drive will operate at said speed, when FOC speed control is active. These results shows operation under FOC-speed control with the machine operating under no-load conditions. An instructive exercise for the reader is to switch operation to FOC

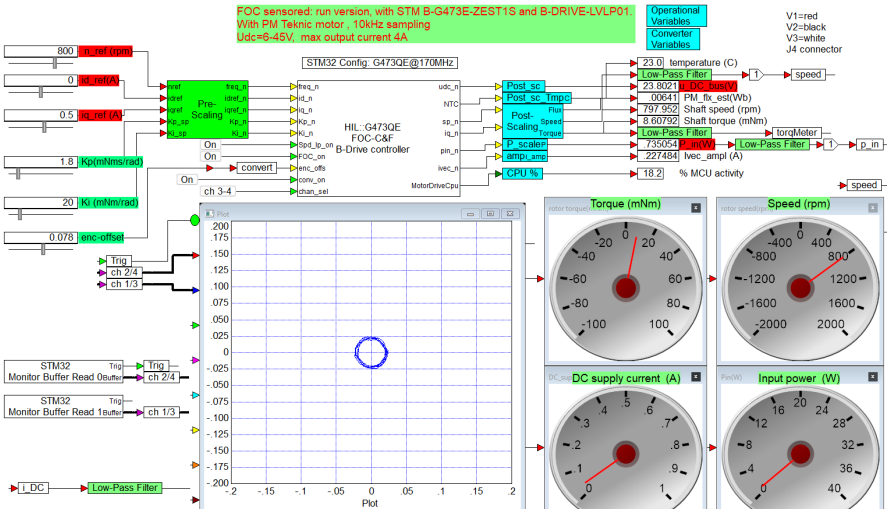


Fig. 4.206 HIL-Debug model, G473E-B-Drive of a PMsyn drive with field-oriented control using an encoder

'torque control' with both direct and quadrature current reference initially set to zero. Then hold the rotor by 'hand' and feel the effect of torque application when moving the quadrature reference slider. Then repeat this exercise by moving the direct axis reference (with the quadrature slider at zero), in which case NO torque on the shaft should be present (assuming that the PM flux is aligned the d-axis of the controller). In this case a slider is present, that can be used to eliminate any encoder offsets. The correct offset slider value is found when operating under torque control and varying the direct axis current reference. Under these conditions, no torque should be produced, as mentioned above.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).

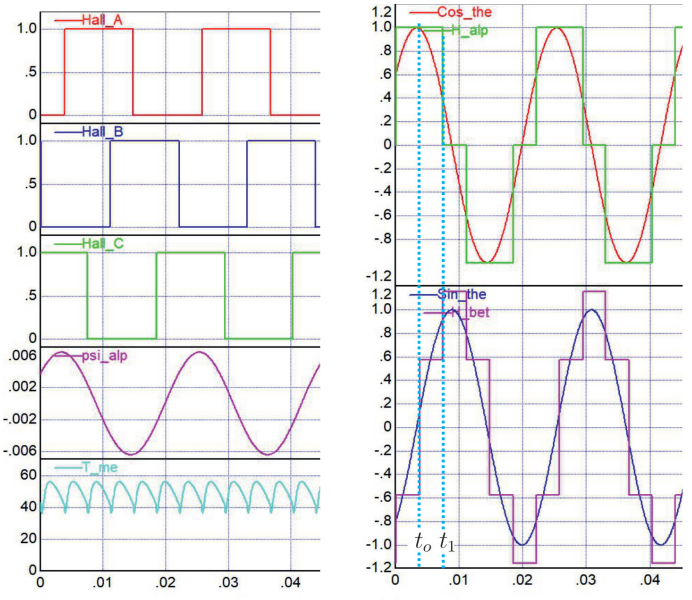
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.4 Sensored Field Oriented Control with Hall sensors

Digital Hall-effect sensors play a key role for brushless DC motor control (BLDC) applications [33], given their simplicity and affordability. In this section the role of said sensors are explored for field-oriented control applications. In this context the results of a simulation model similar to section 4.3.1.1, but adapted to the use of Hall-effect sensors are shown in figure 4.207. Figure 4.207(a) shows the three characteristic rectangular waveforms `Hall_A`, `Hall_B` and `Hall_C`, which are generated by three digital Hall sensors typically working in conjunction with a slotted disc that is locked to the motor shaft. This shows that the signals are intrinsically tied to the flux poles of the machines as is evident by the waveform `psi_alp`, which represents the PM flux vector component $\psi_\alpha = \Re\left\{\vec{\psi}\right\}$. Consequently, one (electrical) 360 deg. rotation of the flux vector corresponds to one period of the Hall signal, as may be observed figure 4.207(a). Also shown in this figure is the shaft torque T_{me}, obtained with the simulation model which shows a torque ripple component.

The three Hall signals are processed by a 'Hall encoder unit' (to be discussed in the `SIM` section) that generates the signals `H_alp` and `H_bet`, which now control the orientation of the 'd,q' coordinate system used by the current controller. For the sake of reference the signals `Cos_the`, `Sin_the` are included, that represent the Cosine and Sine of the PM flux vector angle θ relative to the real-axis. If the Hall signals are correctly aligned relative to the flux,



(a) Hall signals, PM flux and shaft torque

(b) Cos/sin of the electrical angle and processed Hall signals

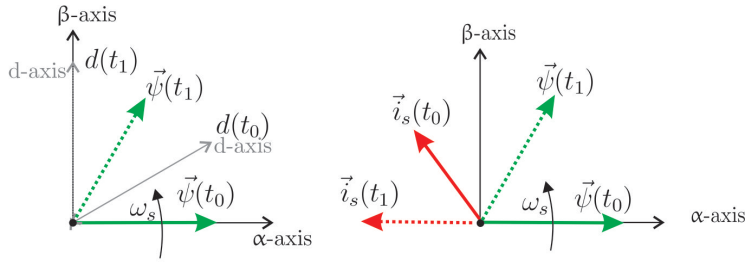
Fig. 4.207 Use of digital Hall sensors for FOC

as is the case here, then the signals `H_alp` and `H_alp` represent a coarse approximation to the ideal Cos/Sin signals `Cos_the`, `Sin_the`, generated by a high resolution encoder. This implies that using Hall-effect signals in effect functions as a low resolution 15 deg (mechanical) encoder.

In terms of understanding the operation of the controller it is instructive to consider the time sequence $t_o \rightarrow t_1$ assuming constant speed operation, equal to ω_s of the PM flux vector shown in figure 4.208. At time mark t_o, the PM flux vector is aligned with the α-axis and the Hall effect signals `H_alp`, `H_bet` define the movement of the d-axis, and its location at time mark t_o (shown as $d(t_o)$) is also shown. As time progresses the flux vector continues to rotates uniformly, when at time mark t_1 the angle between the flux and d-axis vector $d(t_o)$ reaches its upper limit of -30 deg., in which case it makes an instantaneous 60 deg step to its next location $d(t_1)$ as may be observed from figure 4.208(a).

Hence the d-axis vector is 'six-stepping' in the α, β coordinate system.

Likewise, the q-axis vector, which is tied to the current vector $\vec{i}_s$ will also move in 60 deg steps (assuming $i_d^{\text{ref}} = 0$) as shown in figure 4.208(b). Also shown in said figure is the PM flux vector and at time mark t_o the angle between current $\vec{i}_s(t_o)$ and flux vector $\vec{\psi}(t_o)$ is 120 deg, hence a 30 degree offset from the preferred orthogonal location which causes a torque reduction of 14%. As the flux vector continues to move, the angle between the current



(a) vector plot: flux and d-axis (b) vector plot : current and flux

Fig. 4.208 Vector plot: Use of Hall sensors for FOC

vector reduces to 60 deg, at time mark t_1. This explains the torque ripple shown in figure 4.207(a). At this point in time the current vector makes an instantaneous jump to its its next location $\vec{i}_s(t_1)$.

The use of Hall effect signals for FOC is not as effective, when compared to the use of a high resolution encoder, as speed estimation from the six-stepping signals is problematic due to the high harmonic content. However, when considering sensorless operation based on use of the EMF vector, there is a need to identify the flux vector orientation at standstill (even with a ±30 deg max flux error), where a low cost Hall effect sensor is effective and operation with sensorless FOC can be done, as soon as sufficient EMF resolution has been obtained.

4.4.1 Sensored Field Oriented Control with Hall sensors-TI

Purpose of this section is to realize field-oriented control with Hall sensors using a PMsyn or BLDC machine and Texas Instruments hardware. The tool chain, as introduced in figure 2.45 is used as a template for the following subsections, but in this case it is limited to toolchain components: SIM→HIL, for a specific hardware example. For this purpose the `F0049-GANINV` module, connected to a Teknic [2] PMsyn machine has been chosen. However, additional HIL examples (with different TI hardware) will also be provided.

4.4.1.1 Sensored FOC with Hall sensors-F0049-GANINV:SIM

Reference file: `FOC_PM_Hall_F0049-GANINV-SIM.vsm`

This example considers the simulation of a PMsyn Teknic motor [2] connected

to a Texas Instruments GANINV module using Hall sensors, as shown in figure 4.209. The aim is to examine FOC operation of the drive using a fixed point controller, that will subsequently be used in the HIL implementation.

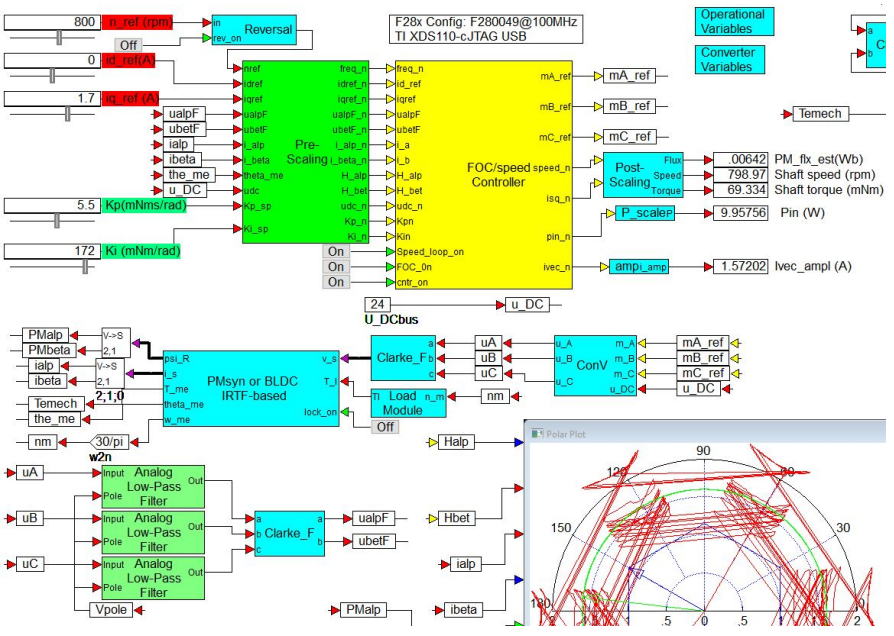


Fig. 4.209 Field-oriented torque/speed control with Hall sensors: simulation

Inputs are the direct and quadrature axis reference currents `id_ref`,`iq_ref` and shaft speed reference `n_ref`. Three logic inputs are shown that provide the following functionality:

- `Ctrl`: converter on/off control.
- `FOC_on`: Field-oriented control active using the Hall sensors. Operation under 'torque control'.
- `Speed_loop_on`: Field-oriented control active using the Hall sensors. Operation under 'speed control'.

When the logic inputs `FOC_on` and `Speed_loop_on` are both OFF, C&F drive control is active, as discussed in section 4.2. Note when operating under torque control, the quadrature axis current reference controls the torque (see equation (2.7a)). The direct axis current reference should be set to zero, when using the Hall effect sensor 'inputs' from the PMsyn machine [2].
When operating under speed control, the speed controller generates a quadrature axis current reference and adjusts torque levels to meet the criteria: measured shaft speed equals user defined reference speed. Also shown in

figure 4.209 are the three LPF modules `Analog Low-pass filter` that are used in the GANINV module to remove the PWM component. Output of the forward Clarke converter variables `ualpF`, `ubetF` are used by the controller to calculate motor input power.
Outputs of the controller are three modulation indices, that control the three converter half bridges. The voltages supplied to the motor `u_A`, `u_B`, `u_C` (see figure 4.176) are found by multiplying the modulation indices `mA_ref`, `mB_ref`, `mC_ref` by half the DC bus voltage as executed in module `ConV`. These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables. The aim of the switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sub chapter 2.1.5). A quadratic load is introduced, which provides a value of 100 mNm at a shaft speed of 1000 rpm. The IRTF based PM model [13], has a user option to select a synchronous or BLDC type machine, as discussed earlier (see figure 2.12). When selecting the 'BLDC' option a sixth harmonic component is added to the PM flux locus. In this case the 'PMsyn' option is selected, given the machine used for this example.

For this example, the `FOC/speed Controller` shown earlier (when operating with an encoder), see figure 4.177, is now used with Hall sensors inputs as indicated in figure 4.207(a). These three Hall signals are forward Clark transformed to stationary coordinate variables `H_alp`,`H_bet` (shown in figure 4.207(b)) by a module `Hall encoder module` located (for simulation) in the `Pre-scaling` unit.

The results shown in figure 4.210, represents operation under torque control, hence the steady state shaft speed is dependant on the applied load torque. Note that at start up ramp functions are used, which determine the rate of reference currents-shaft speed changes, as set in the 'operational Variables' dialogue box.
A comparison of these results with FOC torque control using an encoder (see figure 4.178) is interesting as it shows the presence of torque ripple when using Hall sensors. This is because the use of Hall sensors is comparable with a very low resolution encoder and causes a six-stepping current vector (which is made more complex by the presence of a synchronous current controller with finite bandwidth and gain).

An example with speed control active, as given in figure 4.211, shows that the speed controller current limit 2.5 A (set in the 'Operational Variables' dialogue box) is initially reached. Around time mark ≈120 ms, the reference speed is reached. Correspondingly, the speed controller then provides the quadrature current needed to generate a shaft torque that matches the load torque. A comparison with the FOC-encoder approach, under speed control (see figure 4.179) shows similar results. However, the speed estimator now

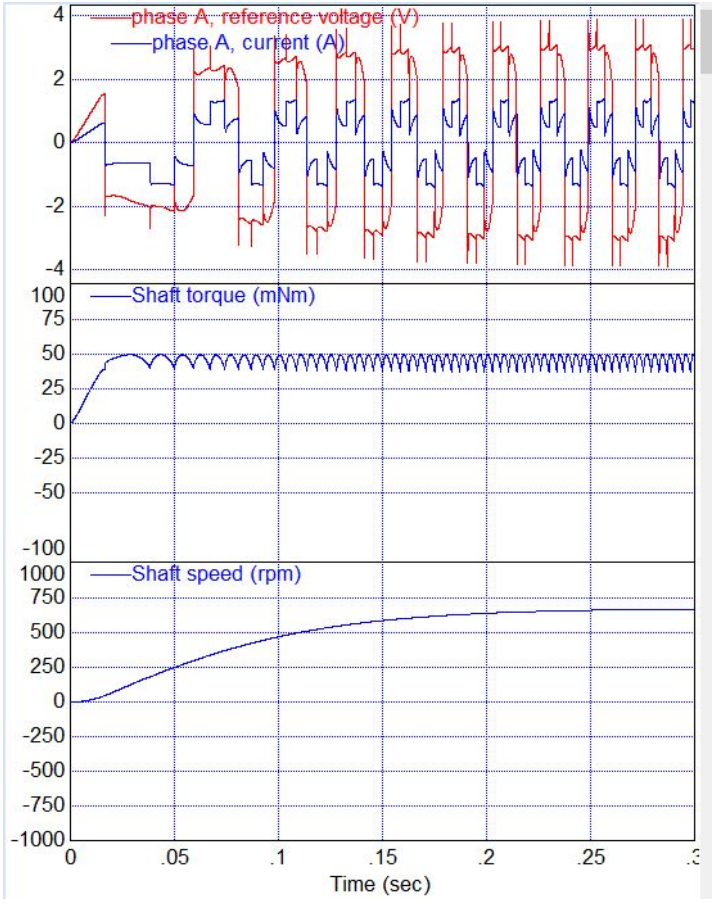


Fig. 4.210 SIM result, PM syn operation, FOC-torque control with Hall sensors active

makes use of Hall sensor variables `H_alp`,`H_bet` and this requires a more demanding low-pass filter (use of a Bessel filter in this case), of which the corner frequency is set in the 'Operational Variables' dialogue box.
Of importance is to consider the vector plot, see figure 4.212 at time mark 300 ms, which shows the orientation of the unity d-axis ('blue'), current vector ('red') and PM flux vector ('green', scaled by a factor 250). Observation of the vector plot confirms the six-stepping nature of the current and d-axis vectors. The vector locus of both should theoretically be hexagonal, but the PI synchronous current controller has finite gain and bandwidth (as mentioned earlier), hence transient effects are clearly apparent (overshoot/undershoot action of the current vector). The orientation of said vectors relative to the flux vector are in accordance with the theoretical basis discussed earlier (see section 4.4).

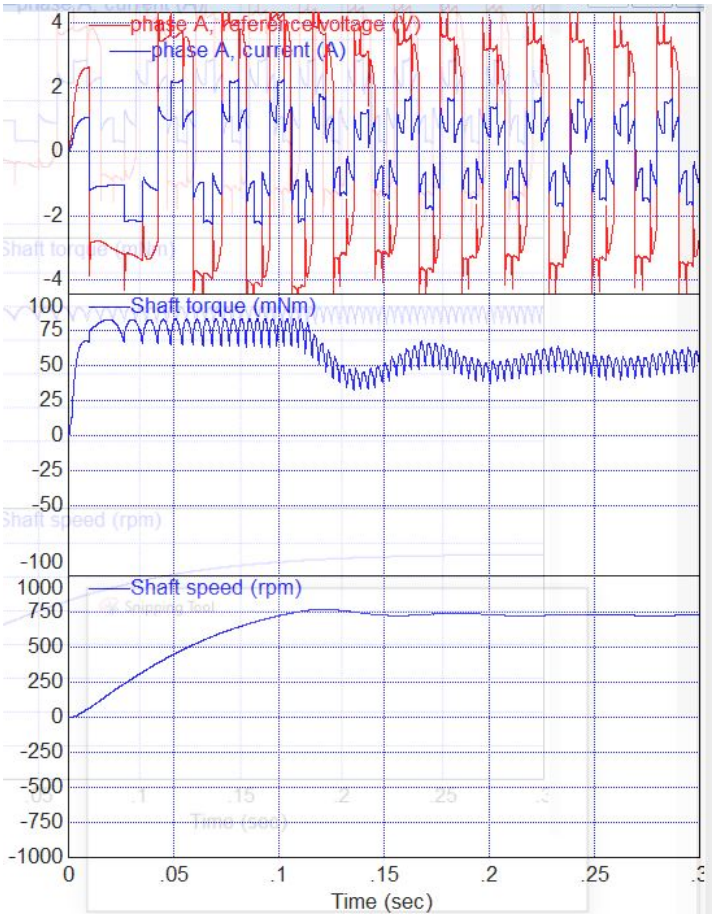


Fig. 4.211 SIM result, PM syn operation, FOC-speed control with Hall sensors active

Operation of the drive is at ≈800 rpm (an error between measured and actual shaft speed is present and caused by the speed estimator). The load determines the quadrature current amplitude, as mentioned earlier.

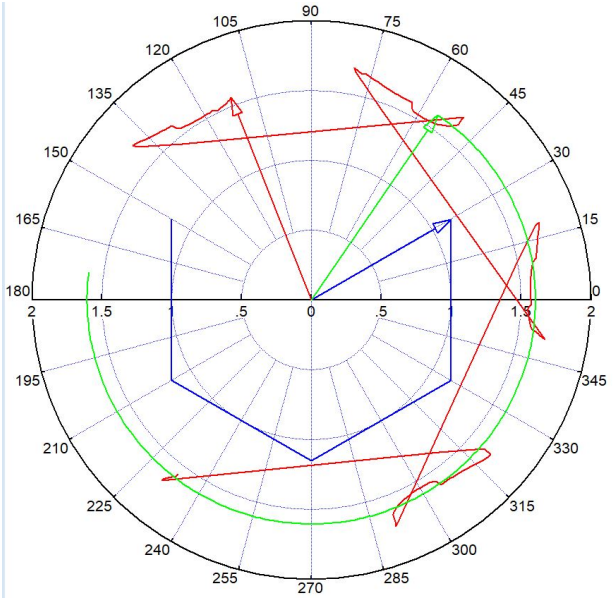


Fig. 4.212 SIM result, PM syn operation, FOC-speed control with Hall sensors: vector plot

4.4.1.2 Sensored FOC with Hall sensors-F0049-GANINV: HIL

Reference files:

- `FOC_sensored_Hall_F0049-GANINV-compile.vsm`, for HIL-compile
- `FOC_sensored_Hall_F0049-GANINV-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.213 represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F280049` module) which is connected to a PMsyn Teknic motor[2].

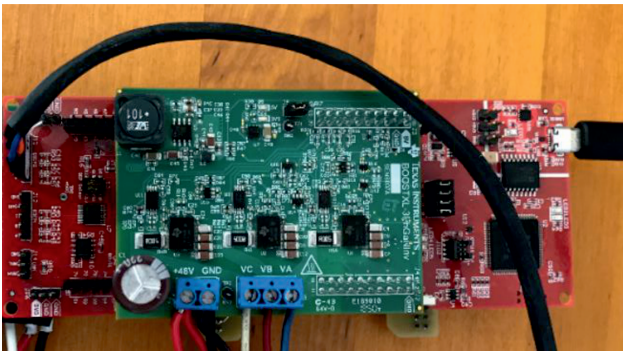


Fig. 4.213 F280049 MCU- GANINV setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.214 shows the 'FOC H-sensored and C&F control' module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct and quadrature axis reference currents `id, iq`, and the reference shaft frequency `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller.
Four logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other two are used to select field-oriented control (FOC) operation `FOC_on` and speed operation under FOC `Spd_ctrl_on`, as discussed

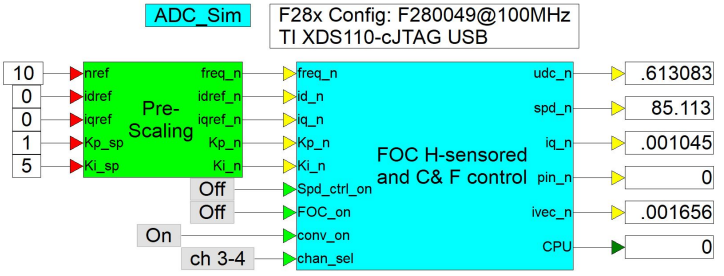


Fig. 4.214 HIL-Compile model of a PMsyn drive with field-oriented control and Hall sensors

earlier (see section 4.3.1.1).
Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- A per unit shaft speed estimate `sp_n` generated via the Hall sensors.
- The measured per unit quadrature current `iq_n`, used to generate a shaft torque estimate.
- A per unit estimate of the motor input power `pin_n`. This is also useful as an estimate for the drive input power (this ignored the converter losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx {}^{p_{\mathrm{in}}}/_{u_{\mathrm{DC}}}$.
- The measured per unit current vector amplitude `ivec_n`.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 4.214 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 4.215.
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box, see figure 4.216, is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference currents-frequency. In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the speed controller sampling frequency is typically set lower than used by

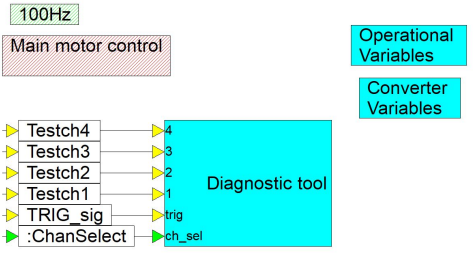


Fig. 4.215 One level down into the FOC Hall sensored Controller module

the main controller. In this case it is set to 1 kHz. Also added to this dialogue module are motor parameters, motor pole pair number and PM flux (as for example found earlier, see the 'Post Analysis' 4.2.1.5).

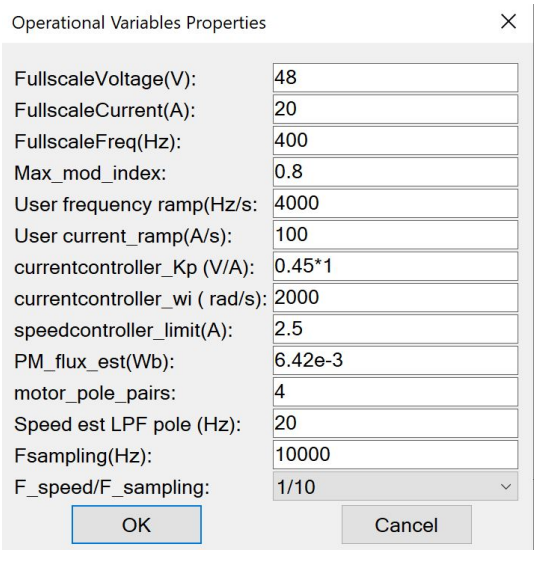


Fig. 4.216 HIL: Dialogue box 'Operational Variables'

The 'Converter Variables' dialogue box, as given earlier (see figure 4.191), defines the scaling used by the Analogue to Digital converter. This module also shows the low-pass filter corner frequency used by the converter for measured the half-bridge voltages (see figure 4.191).
A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1`,

`Test2` or `Test3`, `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.215 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 4.217. Of these, the 'FOC/speed controller' has been discussed earlier (see figure 4.177), where it was used in phases : SIM, PIL and COSIM of the development tool-chain. However, in this case use is made of the Park transformed Hall sensor signals `hall_alp`, `hall_bet` for FOC.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module.
This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F280049` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present, which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after every second (for this drive) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.
The `ADC-PWM-LaunchXL-28004x BOOSTXL-3phGANINV` unit is used generate the PWM signals for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively. ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 4.218 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).

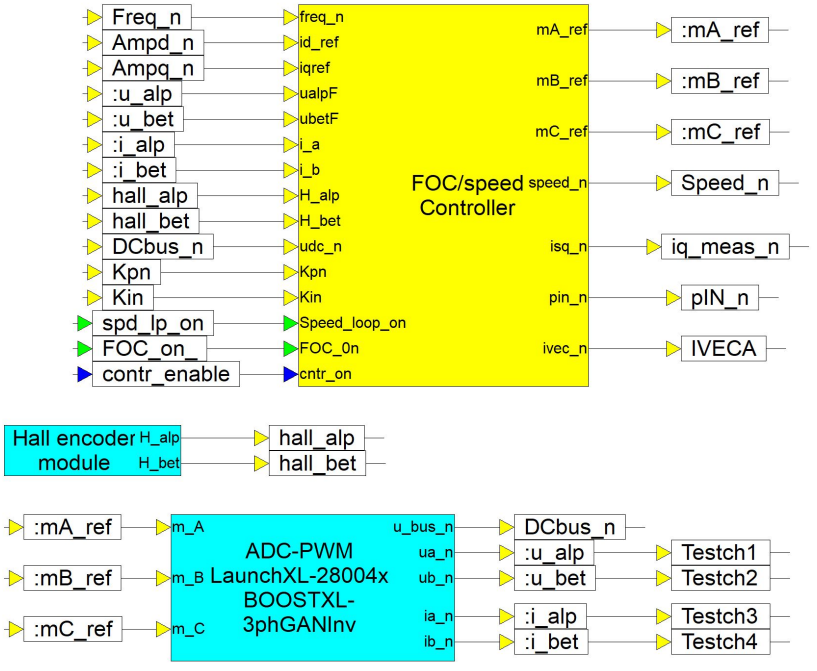


Fig. 4.217 One level down into the 'Main motor controller' module

Central to this figure is the Embed 'target execution module', which executes said out-file. Two sliders are used to set the direct axis current reference value and the electrical frequency. Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Furthermore, a set of numerical display modules have been added, which show: PM flux, estimated shaft speed, estimated shaft torque, estimated input power and current vector amplitude. In addition, a set of gauges has been provided, which show (low-pass filtered) variables: shaft torque, shaft speed, input power and DC supply current.

Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals. The plot given in figure 4.218 displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit voltage vector locus. The electrical frequency reference in figure 4.218 is set to 600 rpm. These results shows operation under FOC-speed control with the machine operating under no-load conditions. An instructive exersize for the reader is to switch operation to FOC 'torque control' (which means 'speed control' OFF) with both direct and quadrature current reference initially set to zero. Then hold the rotor by 'hand' and feel the effect of torque application when moving the quadrature reference slider.

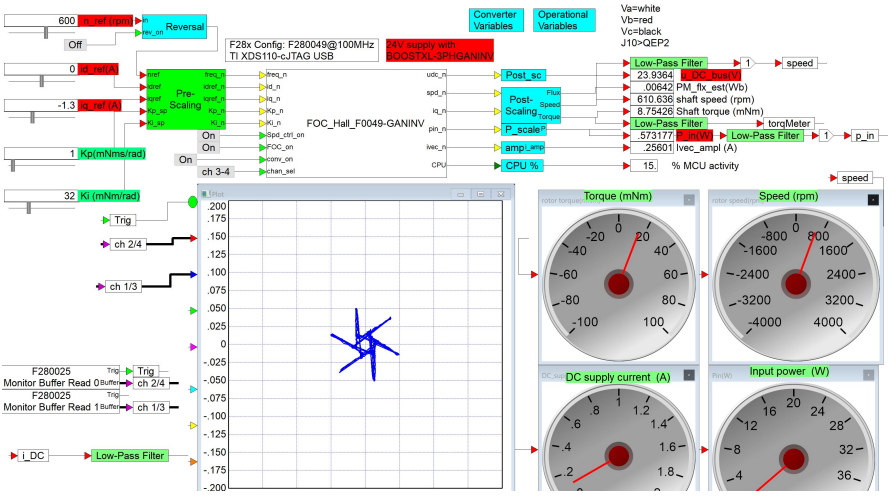


Fig. 4.218 HIL-Debug model, F0049-GANINV of a PMsyn drive with field-oriented control using Hall sensors

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established,

turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.5 Sensored Field Oriented Control with Encoder and Field-weakening

The term 'field-weakening' refers to the introduction of a flux-linkage $\psi_L = -i_d L_s$ component due to the stator inductance (either of the motor itself and/or using external inductances placed in series with the machine). Adding a negative direct axis current component is in opposition to the PM flux (aligned with the d-axis, see figure 4.219).

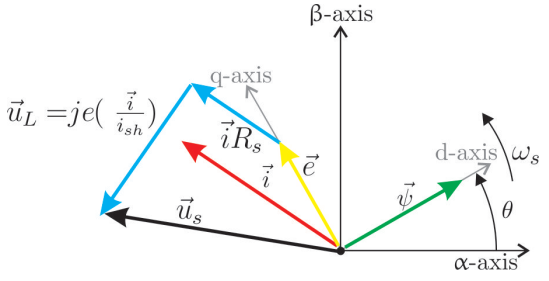


Fig. 4.219 Terminal diagram of PMsyn with field-oriented control and field-weakening using an encoder

The impact of adding a negative direct axis component is that it rotates the current vector $\vec{i}$ into the second quadrant of the (d,q) coordinate system. The key issue is that this current vector rotation rotates the voltage across the stator resistance R_s AND the voltage $\vec{u}_L = j\omega_s L_s \vec{i}$ across the inductance. It is convenient to express this voltage vector in terms of the short-circuit current (as introduced earlier, see equation (2.10)) and repeated here for reader convenience, see equation (4.18).

$$\vec{u}_L = j\left(\frac{\vec{i}}{i_{\text{sc}}}\right)|\vec{e}| \tag{4.18}$$

Rotating the current vector, can have a significant impact on the terminal voltage vector amplitude $|\vec{u}|_s$ as can be observed from figure 4.219. However,

this will **ONLY** be the case if the ratio ($^{i}/_{i_{\mathrm{sc}}}$) is substantial. In other words if the short-circuit current is large relative to the stator current amplitude, the inductance voltage component will be small relative to the EMF amplitude e and field-weakening will not have a significant impact on the terminal voltage vector $|\vec{u}_s|$. If the voltage component $\vec{u}_L$ is substantial, then the terminal voltage amplitude can be reduced and this means that the EMF amplitude can increase, which in turn means that the shaft speed can increase, as this function is defined by expression (4.19).

$$n_m = \frac{30\,|\vec{e}|}{\pi\,\psi\,p} \tag{4.19}$$

The purpose of field-weakening is to extend the shaft speed range of the machine, which is dictated by the choice of DC bus voltage u_{DC} and the maximum modulation index $\hat{m}_{\max}$ of the synchronous current controller, set by the user. Its highest value for linear operation is equal to $\hat{m}_{\max} = {}^{2}/\sqrt{3}$, as discussed earlier (see section 2.1.5). For example, under no-load conditions, the EMF amplitude is equal to the terminal voltage amplitude, in which case the maximum shaft speed is given by expression (4.20).

$$n_m^{\mathrm{noload}} = \frac{15\,\hat{m}_{\max}\,u_{\mathrm{DC}}}{\pi\,\psi\,p} \tag{4.20}$$

For the four pole pair ($p = 4$) Teknic PMsyn motor [2] with a PM flux of 6.42 mWb used in these examples, the maximum no-load speed, using a DC bus voltage of 24 V and $\hat{m}_{\max} = {}^{2}/\sqrt{3}$ is equal to 5152 rpm.

However, the maximum EMF and shaft speed under load conditions and using the inductance, resistance values of the machine will be less for a given terminal voltage amplitude, as may be observed from figure 4.219. Computation of the maximum shaft speed, implies computation of the maximum EMF amplitude for a given terminal voltage amplitude defined in terms of the DC bus voltage and maximum modulation index set by the user.

$$|\vec{u}_s| = \hat{m}_{\max}\frac{u_{\mathrm{DC}}}{2} \tag{4.21}$$

The upper limit for $\hat{m}$ is equal to $2/\sqrt{3}$ (as mentioned above), but care should be taken to ensure that current control remains intact. High modulation index values can cause problems when using shunt resistors in the lower converter switches, as discussed earlier, see figure 2.40. Hence, using in-line current transducers (as is done for example with the GANINV module) is preferred to avoid ADC problems when operating with high modulation indices.

The Embed file `Field-Weakening-calculator-n.vsm` (see figure 4.220), is a convenient tool, which provides the user with an accurate estimate of the maximum shaft speed that can achieved with PMsyn & BLDC machines when operating with field-oriented control. The algorithm used to provide the estimator outputs shown, is directly based on analysis of figure 4.219, where the terminal voltage amplitude is defined by expression (4.21).

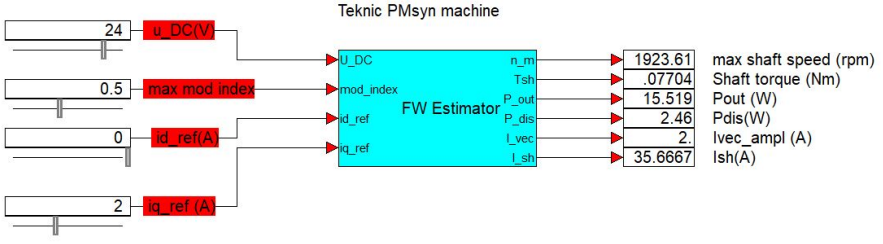


Fig. 4.220 Field-weakening calculator for non-salient PM machines using FOC

The dialogue box for the estimator, shown in figure 4.221, represent the motor parameters, in this case a Teknic machine [2]. Slider inputs are used to specify the DC bus voltage, max modulation index and direct/quadrature current values. After running this example, the output shows the max shaft speed, shaft torque, output power and dissipated power, for FOC. Furthermore, the current vector amplitude `Ivec` and short-circuit current `Ish` are also provided, as these are critical parameters for drives in need of field-weakening.

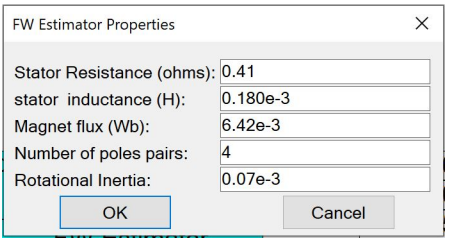


Fig. 4.221 Dialogue box: Field-weakening calculator for non-salient PM machines

An application example for two direct axis values $i_d = 0$, $i_d = -2$A, shown in figure 4.222, underlines the importance of the ratio: max current to short-circuit current (see expression (4.18)).

Variable	FW off: $i_d = 0$ A	FW on: i_d = -2 A
Max shaft speed	1923 rpm	1999 rpm
Shaft torque	77 mNm	77 mNm
Ouput power	15.5 W	16.1 W
Dissipated power	2.46 W	4.9 W
Current vector ampl.	2 A	2.8 A
Short-circuit current	35.6 A	35.6 A

Fig. 4.222 Influence of field weakening for FOC example using Teknic PMsyn

The results show that field-weakening has no significant impact on the maximum shaft speed (for this drive) that can be realised given the high short-circuit current of the machine relative to the motor current used in this example.
Adding, for example, series inductances in the phases, will reduce the short-circuit current and increase the max shaft speed as can be observed by changing (and then running the file) the inductance parameter value in the dialogue box (see figure 4.221).
Also note that variations of the direct axis current can cause torque changes IF the d-axis of the controller is not correctly aligned with the PM flux vector. Furthermore, the users need to be aware that the machine is indeed a non-salient PM, as saliency effects will cause an additional torque and EMF component, as discussed earlier (see figure 2.16).

Finally, note that when operating at high speeds, with field-weakening active, the EMF amplitude can exceed the DC bus voltage. If under those conditions the direct axis current is suddenly removed (or loss of FOC), a problematic drive situation can occur, which may result in sudden braking of the machine and/or destruction of the converter.

4.5.1 Sensored Field Oriented Control with Encoder and Field-weakening-TI

Purpose of this section is to realize field-oriented control with an encoder using a PMsyn or BLDC machine with field-weakening and Texas Instruments hardware. The tool chain as introduced in figure 2.45 is used as a template for the following subsections, but in this case the toolchain: SIM→ HIL, will be considered for a specific hardware example. For this purpose the `F0049-GANINV` module connected to a Teknic [2] PMsyn machine has been

chosen. However, an additional HIL example (with different TI hardware) will also be discussed.

4.5.1.1 Sensored FOC with Encoder and Field-weakening-F0049-GANINV: SIM

Reference file: `FOC_PM_Enc_FW_F0049-GANINV-SIM.vsm`

This example considers the simulation of a PMsyn Teknic motor [2] connected to a Texas Instruments `F0049 GANINV` converter, as shown in figure 4.223. The aim is to examine FOC operation of the drive using a fixed point controller, that will subsequently be used in the HIL implementation.

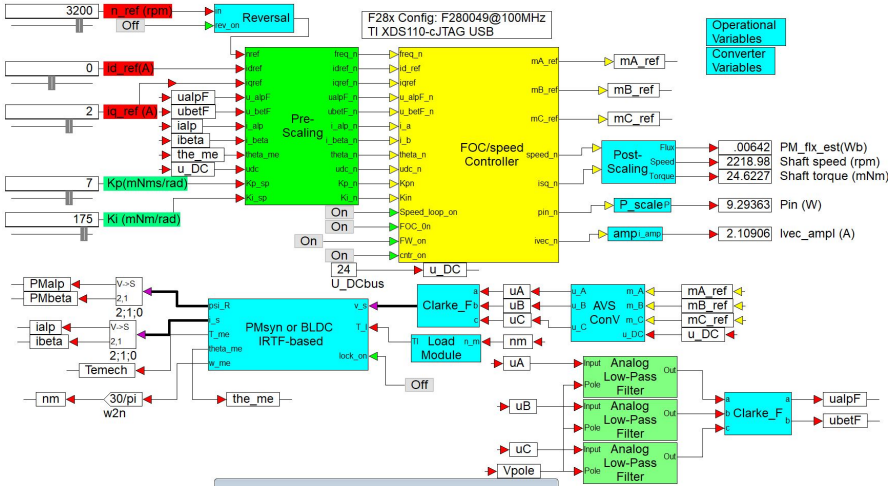


Fig. 4.223 Field-oriented torque/speed control with encoder and field-weakening: simulation

Inputs are the direct and quadrature axis reference currents `id_ref`, `iq_ref` and shaft speed reference `n_ref`. In addition, two sliders are present to set the proportional and integral gain of the speed controller.
Four logic inputs are shown that provide the following functionality:

- `Ctrl`: converter on/off control.
- `FOC_on`: Field-oriented control active using encoder signal. Operation under 'torque control'.
- `Speed_loop_on`: Field-oriented control active using encoder signal. Operation under 'speed control'.

- `FW_on`: Enables the field-weakening controller (and disables the `id` slider input).

When the logic inputs, `FOC_on` and `Speed_loop_on`, are both OFF, drive control is C&F as discussed in section 4.2.
Note when operating under torque control, the quadrature axis current reference controls the torque (see equation (2.7a)). If the PM flux vector is fully aligned with the d-axis (as is the case here), changes to the direct axis current reference will not effect the torque.
When operating under speed control, the speed controller generates a quadrature axis current reference and adjusts torque levels to meet the criteria that the measured shaft speed equals user defined reference speed. When field-weakening is active, the direct current reference is controlled by the field-weakening controller that becomes active when operating near the maximum modulation index value set by the user.

Also shown in figure 4.223 are the three LPF modules `Analog Low-pass filter` that are used in the GANINV module to remove the PWM component. Output of the forward Clarke converter variables `ualpF`, `ubetF` are used by the controller to calculate motor input power.

Outputs of the controller are three modulation indices, that control the three converter half bridges. The voltages supplied to the motor `u_A`, `u_B`, `u_C` (see figure 4.223) are found by multiplying the modulation indices `mA_ref`, `mB_ref`, `mC_ref` by half the DC bus voltage as executed in module `AVS ConV`. These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables. The aim of the switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sub chapter 2.1.5). A quadratic load is introduced, which provides a value of 20 mNm at a shaft speed of 2000 rpm.

The IRTF based PM model [13] has a user option to select a synchronous or BLDC type machine, as discussed earlier (see figure 2.12). When selecting the 'BLDC' option a sixth harmonic component is added to the PM flux locus. In this case the 'PMsyn' option is selected, given the machine used for this example.

Drilling into the `FOC/speed Controller` reveals the structure shown in figure 4.224.
Readily apparent is the synchronous current controller module, as discussed earlier, see figure 4.125. Both proportional gain, bandwidth and the maximum modulation index of said controller must be provided in the 'Operational variables' dialogue box. The speed controller module, as introduced earlier (see section 3.3), must be provided with appropriate proportional and integral gain values, that depend on the combined load inertia and user defined speed control bandwidth (see equation (2.4)).

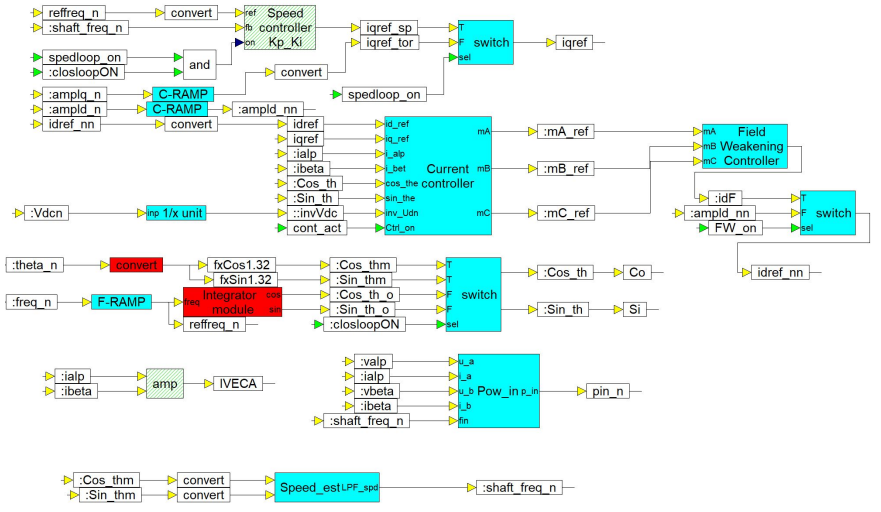


Fig. 4.224 FOC/speed controller structure with Field-Weakening

A set of sliders is introduced in figure 4.223 for this purpose. Added to this controller is a `Field Weakening Controller` (FW), which makes use of the three modulation indices. The FW controller compares the maximum instantaneous modulation index value with the reference (set by the user) and a negative direct axis current (limited to an absolute max value) is generated when the reference is reached.

An example with speed control active, given in figure 4.225, shows that the speed controller current limit 2.5 A (set in the 'Operational Variables' dialogue box) is initially reached and around the time mark 150 ms the modulation index reaches 0.6 (which corresponds to a terminal vector amplitude of 6 V) is reached and negative direct axis current is increased till the user defined maximum value is reached.
As speed increases, the EMF increases to the point (at approximately time mark 170 ms) when q-axis current controller operation is compromised and q-axis current is reduced, which leads to the shaft torque reduction shown. In this example the shaft speed does NOT reach the reference speed value set by the user.

Of importance is to consider the vector plot (see figure 4.226) at time mark: 500 ms, which shows the orientation of the unity d-axis ('green'), voltage vector ('blue'), PM flux vector ('brown', scaled by a factor 500) and the current vector ('red'). Observation of the current vector confirms presence of a large 2 A direct axis current component. The terminal voltage amplitude

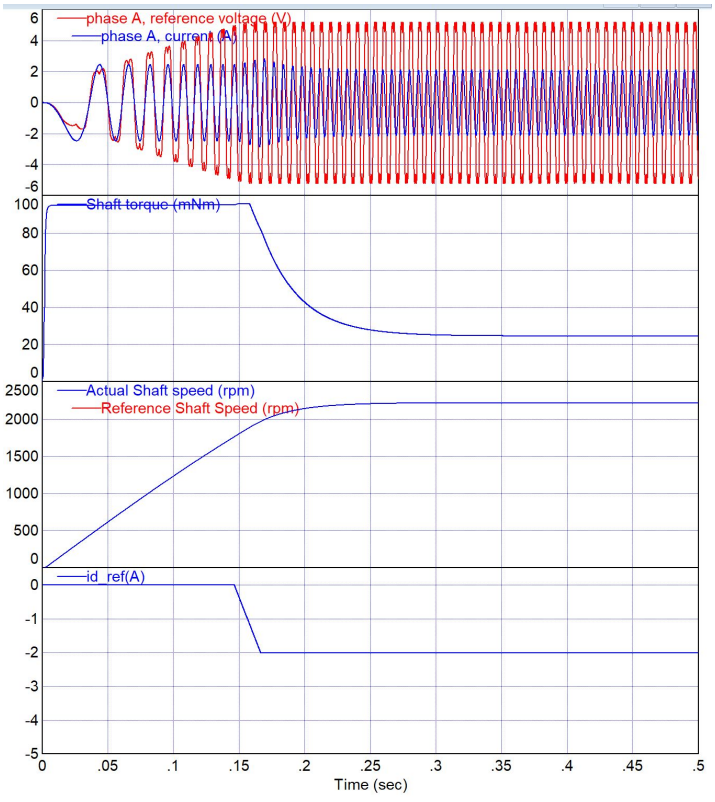


Fig. 4.225 SIM result, PM syn operation, FOC-speed control and field-weakening active

is indeed equal to 6 V as expected.
The impact of field-weakening on the maximum achievable shaft speed is limited because the motor current amplitude is low compared to the short-circuit current of the machine.

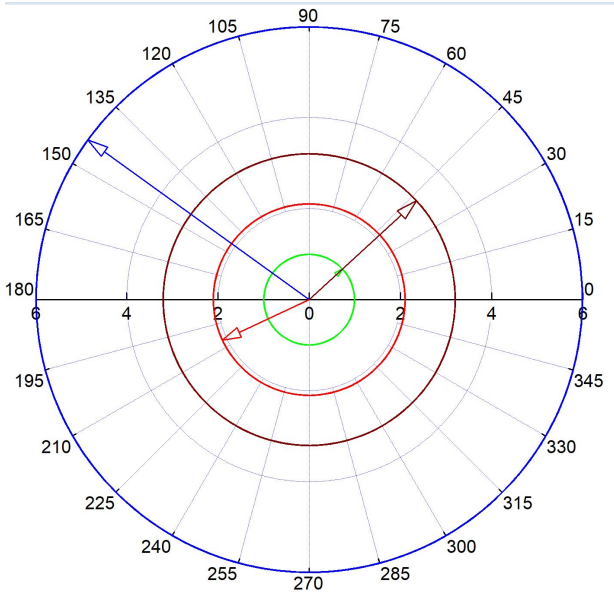


Fig. 4.226 SIM result, PM syn operation, FOC-speed control & FW active: vector plot

4.5.1.2 Sensored FOC with Encoder and Field-weakening-F0049-GANINV: HIL

Reference files:

- `FOC_sensored_Enc_F0049-GANINV&FW-compile.vsm`, for HIL-compile
- `FOC_sensored_Enc_F0049-GANINV&FW-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.227 represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F280049` module), which is connected to a PMsyn Teknic motor [2].

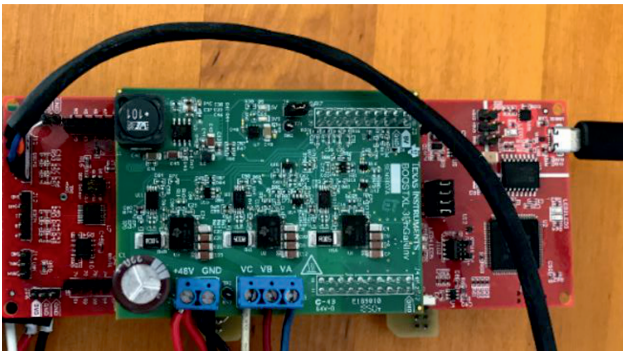


Fig. 4.227 F280049 MCU-GANINV setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. Furthermore, when using an encoder the motor connections should be precisely as specified in the 'debug' file, to ensure correct alignment between the controller d-axis and the PM motor flux vector.
A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.228 shows the 'FOC sensoredwith FW and C&F control' module, which must be compiled to generate an out-file. Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct and quadrature axis reference currents `idref`,`iqref`, and the reference shaft frequency `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller.

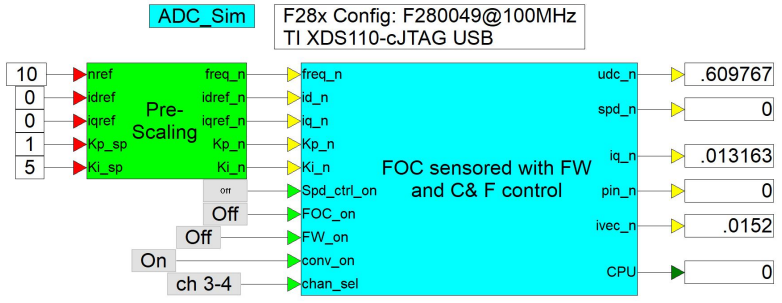


Fig. 4.228 HIL-Compile model of a PMsyn drive with field-oriented control/encoder and field-weakening option

Five logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other three are used to select field-oriented control (FOC) operation `FOC_on`, speed operation using FOC `Spd_ctrl_on` and activating the field-weakening option using `FW_on`. Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- A per unit shaft speed estimate `sp_n` generated via the encoder.
- The measured per unit quadrature current `iq_n`, used to generate a shaft torque estimate.
- A per unit estimate of the motor input power `pin_n`. This is also useful as an estimate for the drive input power (but this ignores the converter losses).
- The measured per unit current vector amplitude `ivec_n`.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 4.228 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Control module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes shown in figure 4.229.
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.230), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current(s)-frequency.

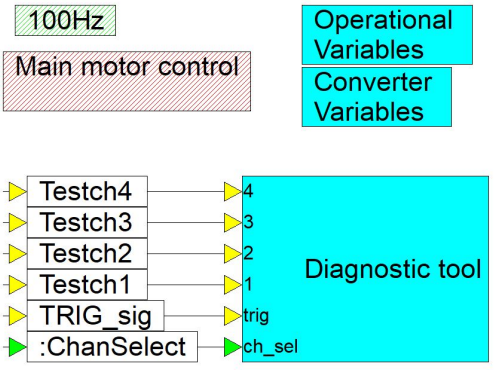


Fig. 4.229 One level down into the FOC sensored Controller module

In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive.

Furthermore, the speed controller sampling frequency is typically set lower then used by the main controller. In this case it is set to 1 kHz. Also added to this dialogue module are motor parameters: motor pole pair number and PM flux (as for example found earlier, see the 'Post Analysis' 4.2.1.5). The entry `max_Id`, defines the maximum (absolute) negative direct axis for the field-weakening controller.

The 'Converter Variables' dialogue box, introduced earlier, see figure 4.191, defines the scaling used by the Analogue to Digital converter, as discussed earlier (see equation (4.2)).
A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.229, is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit.
Moving one level lower into the 'Main motor Control' module reveals a set

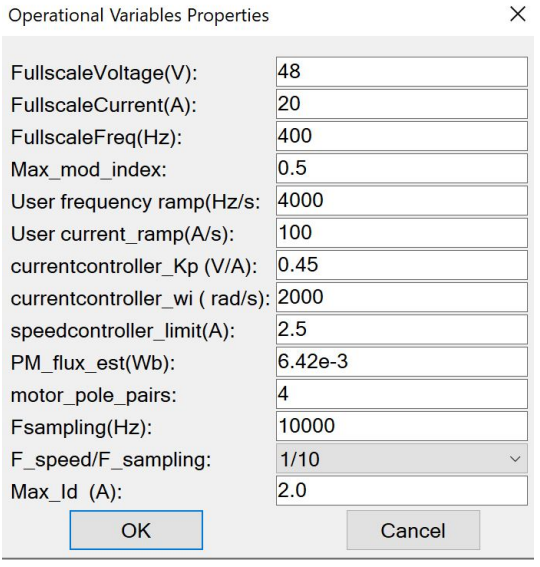


Fig. 4.230 HIL: Dialogue box 'Operational Variables'

of modules, shown in figure 4.231. Of these, the 'FOC/speed controller' has been discussed earlier (see figure 4.177), where it was used in phases : SIM, PIL and COSIM of the development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module.
This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F280049` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present, which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after every second (for this drive) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.
The `ADC-PWM-LaunchXL-28004x BOOSTXL-3phGANINV` unit is used generate the PWM signals for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively. Field-weakening control is realised with variable `FW_on_`, which controls the direct axis current when active (see figure 4.224).

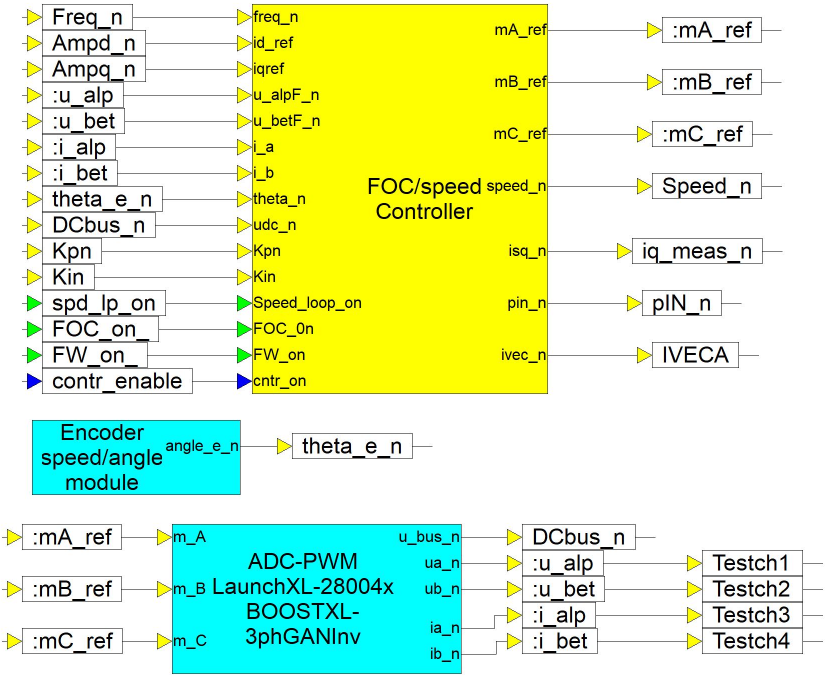


Fig. 4.231 One level down into the 'Main motor controller' module

ADC offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 4.232 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct/quadrature axis current reference values and the shaft speed. An additional two sliders are used to set the proportional and Integral gain of the speed controller.
Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Furthermore, a set of numerical displays are present which show: PM flux, shaft speed, shaft torque, input power and current vector amplitude. In addition, a set of gauges are present which monitor (low-pass filtered): shaft torque, shaft speed, input power and DC supply current.

Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

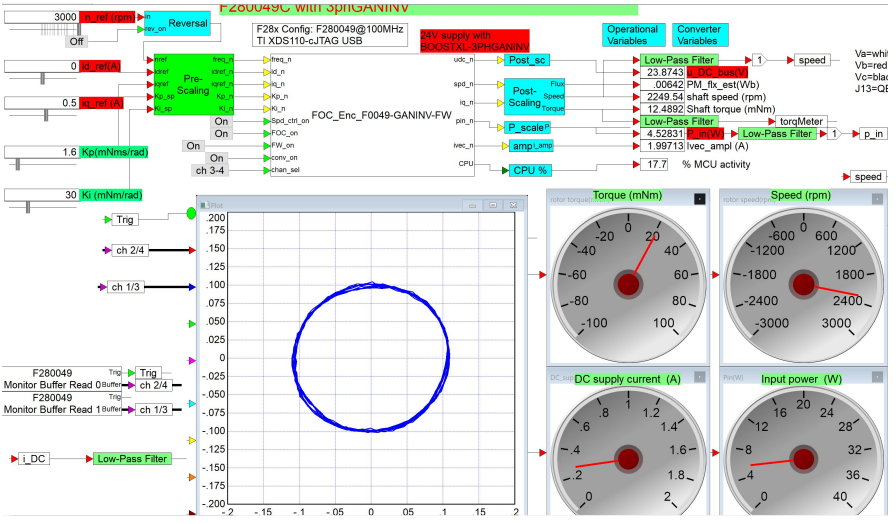


Fig. 4.232 HIL-Debug model, F0049-GANINV of a PMsyn drive with field-oriented control/encoder and field-weakening option

The plot given in figure 4.232, displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit LPF voltage vector locus. The shaft speed reference in figure 4.232 is set to 3000 rpm.
These results shows operation under FOC-speed control with the machine operating under no-load conditions and field-weakening option active. Observation of these results show that the drive is operating with maximum negative direct axis current, in this case set to 2 A, which allows operation at a maximum shaft speed of 2249 rpm.
With field-weakening `off` the maximum shaft speed is 2157 rpm, as anticipated, given that the short-circuit current is high relative to the peak vector current amplitude.
When operating with field-weakening active the per unit terminal voltage amplitude will be ≈0.125 (as can be observed by selecting `ch 1-2`), given that the maximum modulation index for the synchronous controller is set to 0.5 for this example.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.5.1.3 Sensored FOC with Encoder and Field-weakening-F069-boostXL8301: HIL

Reference files:

- `FOC_sensored_Enc_F069-8301&FW-compile.vsm`, for HIL-compile
- `FOC_sensored_Enc_F069-8301&FW-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.233 represents the `BOOSTXL8301` converter
(located on the `LAUNCHXL-F28069` module), which is connected to a PMsyn Teknic motor [2].
For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. Furthermore, when using an encoder the motor connections should be precisely as specified in the 'debug' file, to ensure correct alignment between the controller d-axis and the PM motor flux vector.
A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given

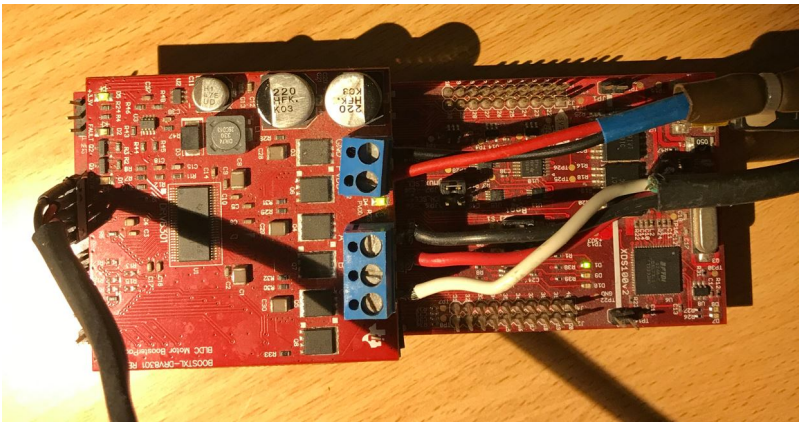


Fig. 4.233 F28069 MCU-BoostXL8301 setup for HIL operation

controller configuration. The model given in figure 4.234 shows the 'FOC sensored with FW and C&F control' module, which must be compiled to generate an out-file. Inputs, shown in the stage as constants (for readability

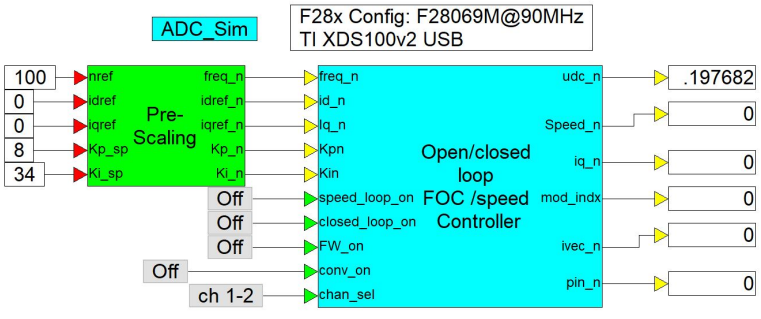


Fig. 4.234 HIL-Compile model of a PMsyn drive with field-oriented control/encoder and field-weakening option

purposes) for this controller module are the user direct and quadrature axis reference currents `idref`,`iqref`, and the reference shaft frequency `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller. Five logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other three are used to select field-oriented control (FOC) operation `FOC_on`, speed operation using FOC `Spd_ctrl_on` and activating the field-weakening option using `FW_on`.
Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- A per unit shaft speed estimate `speed_n` generated via the encoder.
- The measured per unit quadrature current `iq_n`, used to generate a shaft torque estimate.
- The modulation index `modindx`, used to calculate the voltage vector amplitude.
- A per unit estimate of the motor input power `pin_n`. This is also useful as an estimate for the drive input power (bus this ignores the converter losses).
- The measured per unit current vector amplitude `ivec_n`.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 4.234 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes shown in figure 4.235.

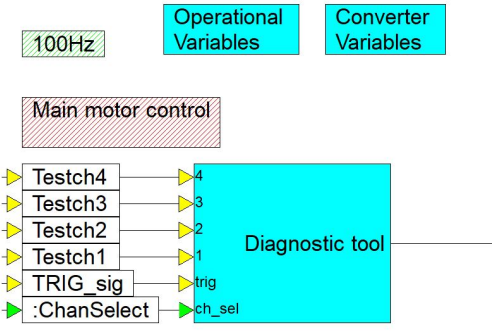


Fig. 4.235 One level down into the FOC sensored Controller module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.236), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current(s)-frequency (shaft speed). In addition, the speed controller quadrature current (absolute value)

has been added, which in effect sets the shaft torque limit for the drive. The entry `max_Id`, defines the maximum (absolute) negative direct axis for the field-weakening controller.

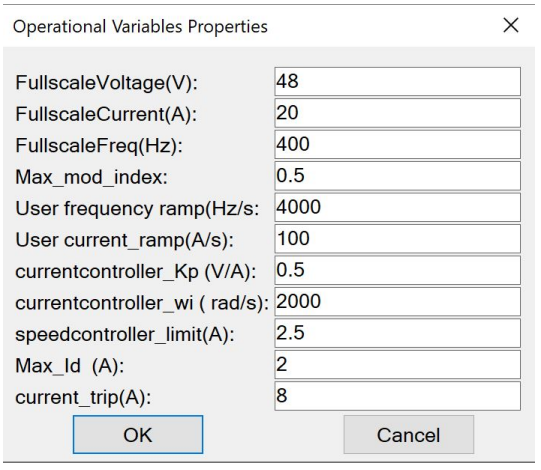


Fig. 4.236 HIL: Dialogue box 'Operational Variables'

The 'Converter Variables' dialogue box (see figure 4.237), defines the scaling used by the Analogue to Digital converter, as discussed earlier (see equation (4.2)).

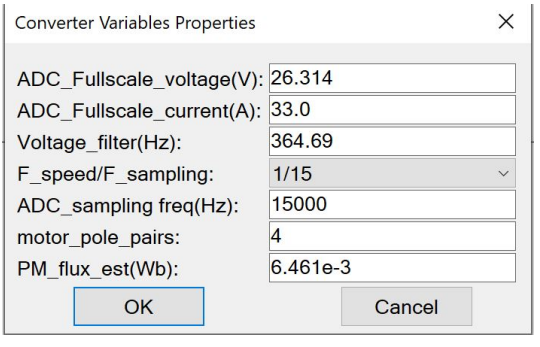


Fig. 4.237 HIL: Dialogue box 'Converter Variables'

Also added to this dialogue module are motor parameters: motor pole pair number and PM flux (as for example found earlier, see the 'Post Analysis' 4.2.1.5). Furthermore, the speed controller sampling frequency is typically set lower then used by the main controller. In this case it is set to 1 kHz.

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 4.235 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit.
Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 4.238. Of these, the 'FOC/speed controller' has been discussed earlier, see figure 4.177 where it was used in phases : SIM, PIL, COSIM, HIL of the development tool-chain for a different MCU.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 30 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: 0.1 μs
- Board in use: `LAUNCHXL-F28069` with `BOOSTXL-8301` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after every second (for this drive) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.
The `ADC-PWM-LaunchXL-069 BOOSTXL2` unit is used generate the PWM signals for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively. Field-weakening control is realised with variable `FW_on`, which controls the direct axis current when active (see figure 4.238).
ADC offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage `DCbus_n_n`, completes the ADC measurement sequence.

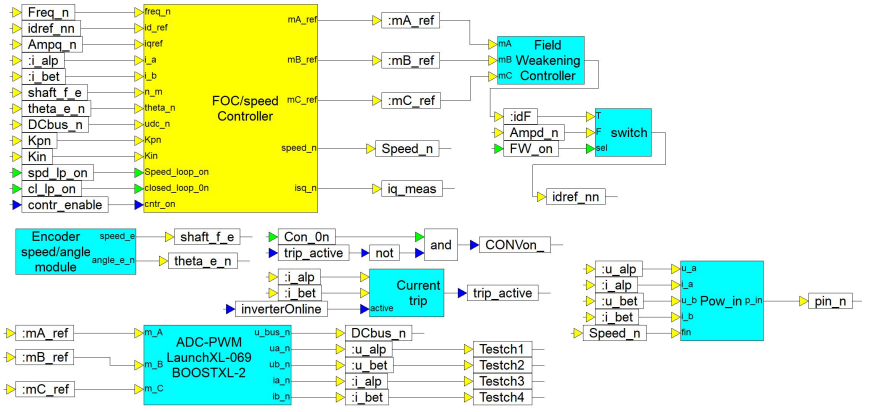


Fig. 4.238 One level down into the 'Main motor controller' module

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 4.239 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct axis current reference value and the shaft speed reference. In addition, two slider are added for setting the proportional and integral speed gain of the speed controller.
Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Furthermore, a set of numerical displays are present which show: PM flux, shaft speed, shaft torque, input power, current vector amplitude and voltage vector amplitude. In addition, a set of gauges are present to monitor (low-pass filtered): shaft speed, shaft torque and input power.
Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.
The plot given in figure 4.239 displays the per unit LPF measured voltage vector locus. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit measure current vector locus. The shaft speed reference in figure 4.232 is set to 2500 rpm.
These results show FOC-speed control with the machine operating under no-load conditions and field-weakening option active.
Observation of these results learns that the drive is operating with maximum negative direct axis current in this case set to 2 A (see numeric display `i_vec(A)`), which allows operation at a maximum shaft speed of 2242 rpm.
When operating with field-weakening active the per unit terminal voltage amplitude will be ≈0.125 (as can be observed by selecting `ch 1-2`), given that the maximum modulation index for the synchronous controller is set to 0.5 for this example.

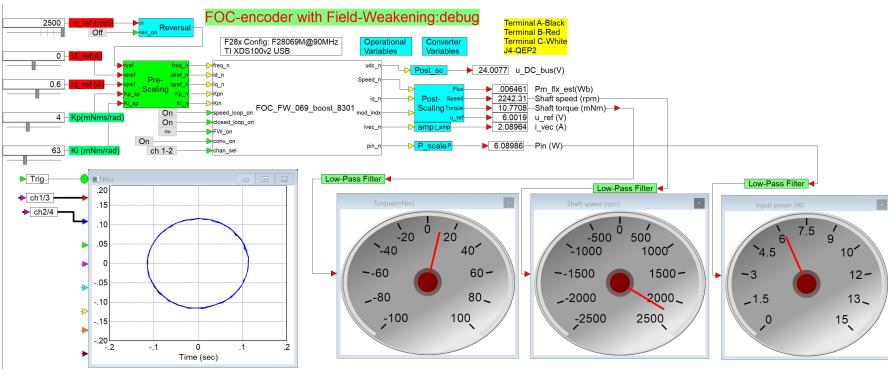


Fig. 4.239 HIL-Debug model, F069-boostXL-8301 of a PMsyn drive with field-oriented control/encoder and field-weakening option

Note that care should be taken when operating at the highest allowable modulation index of $2/\sqrt{3}$, as under these conditions the **phase** modulation index amplitude will be equal to one. For converters using current shunts in the lower switches (as is the case here) this can become problematic, as a situation will then occur where there is insufficient time for the ADC to measure the currents. This will compromise current control, unless action is take to remedy this problem.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.6 Sensorless Field Oriented Control PMsyn & BLDC

The term 'sensorless' specifically refers to the exclusion of the shaft sensor normally used to achieve field-oriented control of a PM drive. When using an encoder, the aim is to align the controller d-axis with the rotor flux vector, as shown in figure 4.240.

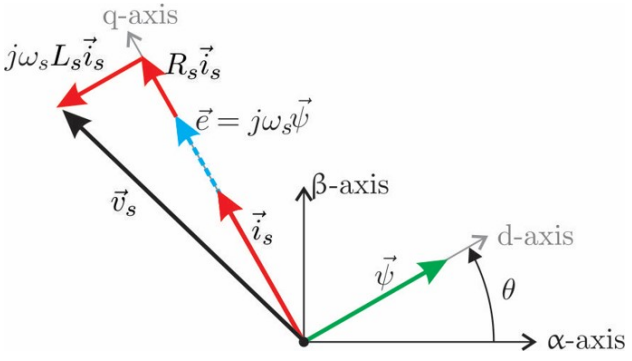


Fig. 4.240 Terminal vector plot: FOC operation PMsyn & BLDC drives

where $\theta = \omega_s\, t$ represents the orientation of the d,q reference frame relative to the stationary α, β coordinate system. The term 'sensorless' is therefore more appropriately called 'encoderless' field-oriented control (FOC).

Encoderless FOC control is principally based on identification of the EMF vector $\vec{e}$ in terms of its amplitude and orientation in the α, β coordinate system. Once this is established the rotor flux is estimated by integrating said EMF vector, using:

$$\vec{\psi}^{\text{est}} = \frac{\vec{e}}{j\omega_s} \tag{4.22}$$

Sensorless field-oriented control is then established by aligning the controller d-axis with the estimated rotor flux vector.

Equation (6.16) shows an important issue with respect to rotor flux estimation. When the electrical frequency moves from $\omega_s \rightarrow 0$ the EMF amplitude will likewise move: $|\vec{e}| \rightarrow 0$. Which implies that expression (6.16) becomes undefined at zero stator frequency, i.e at zero shaft speed of the PMsyn or BLDC. Consequently, an encoderless algorithm such as 'AFT' or 'FAST' [23] must be able to handle this issue.

To achieve 'good' encoderless control, the following guidelines should be adhered to:

- Accurate measurement of the terminal voltage vector $\vec{v}_s$. Here use is made of the low-pass filtered converter voltages which are then Clarke transformed. The use of low-pass filtered voltages, implies the need for accurate knowledge of the LPF corner frequency implemented in the converter, as this information must also be passed to the sensorless algorithm.
- Accurate measurement of the current vector $\vec{i}_s$. When using current shunts in the lower converter switches, current measurements when operating at high modulation index values, e.g at high speeds, may be compromised and this can affect EMF estimation. Using phase 'in-line' current shunts/transducers is therefore preferred for sensorless FOC.
- Compensation of ADC offsets for voltage and current measurement.
- Optimum utilization of the typically 12bit ADC converter(s), which in effect, means matching the the ADC-fullscale voltage/current settings of the drive to the application as discussed earlier, see section (4.1.1.3).
- Accurate stator resistance estimation: As the shaft speed reduces the estimated EMF vector is dominated by the term $\left(\vec{u}_s - R_s\,\vec{i}_s\right)$. In this context an important figure of merit was introduced earlier and referred to as the 'EMF threshold speed' n_e (see equation (2.12a)). This is the shaft speed at which the EMF amplitude equals the voltage drop across the resistance and this can also be expressed as a ratio $\left(\frac{n_e}{T_{\mathrm{me}}}\right)$, where T_{me} represents the shaft speed and may be calculated using expression (4.23).

$$\frac{n_{\mathrm{e}}}{T_{\mathrm{me}}} = \frac{20\,R_s}{\pi\;p^2\psi_R^2}\,(\mathrm{rpm/Nm}) \tag{4.23}$$

Hence accurate voltage/current measurements AND accurate stator resistance values are needed, particularly at start-up of the drive. Which is why

resistance estimation, using DC current injection is preferred at that time, as it also aligns the controller d-axis with the flux vector.
- Accurate estimation of the stator inductance L_s: As shaft speed increases (and load) the voltage across the inductance becomes an important term for EMF estimation as may be observed from figure 4.240. The importance of this term is dependant on the ratio quadrature current/short circuit current, as discussed earlier (see equation (2.10)).
- Good knowledge of the machine PM rotor flux: A sensorless algorithm requires access to an accurate estimate of the rotor flux. This estimate may be found by operating under current & frequency control conditions, see section 4.2.1.5.

One of the important issues mentioned above, is the need for accurate estimated values for the stator resistance $\tilde{R}_s$ and stator inductance $\tilde{L}_s$. Ideally these must match the actual values R_s, L_s of the motor. Any errors between estimated and actual parameters will cause an error between the estimated EMF $\tilde{e}$ and actual EMF e of the machine.

The impact of such errors are shown with the aid of equation (4.24).

$$\tilde{e} = e\left(1 + \left(\frac{i_d}{i_{\text{sc}}}\right)\left(1 - \frac{\tilde{L}_s}{L_s}\right)\right) \text{ no} - \text{load operation, } i_q = 0 \quad (4.24a)$$

$$\tilde{e} = e\left(1 + \left(\frac{i_q R_s}{e}\right)\left(1 - \frac{\tilde{R}_s}{R_s}\right)\right) \text{ load operation with, } i_d = 0 \quad (4.24b)$$

These expressions are found using figure 4.240 under the conditions specified above. The impact of parameter errors can, with the aid equation (4.24), be outlined as follows:

- Operation under no-load ($i_q = 0$): Expression (4.24a) shows the impact of an error between the estimated $\tilde{L}_s$ and actual motor inductance L_s.
  If $\tilde{L}_s = L_s$ then the estimated EMF $\tilde{e}$ equals the actual EMF e and therefore the estimated flux and actual flux will also be the same. If $\tilde{L}_s \neq L_s$, then there will be a difference between estimated and actual EMF's. However, the difference will depend on the ratio of the applied direct axis current and short-circuit current of the machine, defined as $\left(\frac{i_d}{i_{\text{sc}}}\right)$. If this ratio is relatively low, then errors between estimated and actual inductance will have less impact on the estimated EMF.
- Operation under load (with $i_d = 0$): Expression (4.24b) shows the impact of an error between the estimated $\tilde{R}_s$ and actual motor resistance R_s.

If $\tilde{R}_s = R_s$ then the estimated EMF $\tilde{e}$ equals the actual EMF e and therefore the estimated flux and actual flux will also be the same. If $\tilde{R}_s \neq R_s$, then there will be a difference between estimated and actual EMF's. However the difference will depend on the ratio of the voltage across the resistance $(i_q\, R_s)$ and EMF e of the machine, defined as $\left(\frac{i_q\, R_s}{e}\right)$. This ratio is a function of the shaft speed and for the case where the shaft speed equals the EMF threshold speed n_e (see equation 2.12) it will be 1. Consequently, when operating above the EMF threshold speed, i.e $n_m > n_e$ the impact of stator resistance errors will be reduced as apparent from equation (4.24b). When operating at shaft speeds below the EMF threshold speed and in the zero speed speed region the impact of resistance errors will be large and can be problematic if $\tilde{R}_s > R_s$, as this can lead to the case $\tilde{e} = -e$,i.e the estimated EMF is such that it is operating with negative speed. Hence, the use of estimating the stator resistance at start-up is advisable to avoid such issues.

The benefits of using encoderless FOC (when compared to using an encoder) are as follows:

- No need for adding an encoder to the electrical machine, which reduces (by typically 30%) the overall space needed for the drive and reduces the amount of cabling to the machine (no encoder cable, just the three phase wires).
- In applications where both shaft ends of the motor are required the use of an encoder can be problematic.
- For applications where reliability of drive operation is critical, the use of encoderless FOC is often used as a 'backup'. For example, encoder failures can then be bypassed by moving to sensorless FOC, hence ensuring continuity of operation.
- Commissioning of a drive with an encoder requires careful consideration of the terminal connections, otherwise misalignment between the rotor flux vector and controller d-axis can occur.
- Misalignment errors between the rotor flux vector and the controller d-axis caused by not orientating the encoder correctly, are avoided when using sensorless FOC.
- Sensorless FOC functions as a high resolution encoder, which improves controllability of the drive.
- The cost issue: the removal of the encoder reduces the overall cost of the drive.

The advantages of using sensorless FOC for PMsyn & BLDC must be weighed up against the issues mentioned in the 'guidelines' above. Applications that require operation in the zero speed region (particularly when high torque levels over an extended time period are required), may need additional control

measures to maintain flux alignment with the controller d-axis in said zero speed region. Alternatively, use can be made of an IPM machine, when the saliency component can be used to achieve alignment between the flux vector and d-axis at zero speed.

4.6.1 Sensorless Field Oriented Control using AFT®-STM

The examples linked to ST Micro will be covered using the development toolchain components: SIM→HIL, based on the use of the F303. This MCU wil be used to run 'Altair Flux Tracker' (AFT®), which is a classified sensorless algorithm AFT®and access to said algorithm is by way of an Embed DLL. Initially the tool-chain components presented in the following subsections will be based on the use of the F303-ESC001 drive setup with a BLDC drone [8].

4.6.1.1 AFT FOC-F303-ESC001:SIM

Reference file:

- `DLL-AFT_F303CB_ESC100_Drone_Sim.vsm`

The simulation (SIM) approach discussed in this subsection (see figure 4.241), is designed to evaluate the controller within the target MCU, in this case the F303, which contains the DLL used to communicate with AFT®.
The module `Main motor Controller(Sim)`, generates the modulation indices required for the converter/motor. In this example the IRTF machine model is set to operate in BLDC mode (not PMsyn, given the use of a BLDC machine). The three LPF modules, are used to model the converter R-C network used to measure the motor voltage vector.
Moving one level into the module `Main motor Controller(Sim)`, shows a 'FOC/speed Controller-AFT' (see figure 4.242).
Added to said controller are two modules responsible for input power measurement `Pow_in` and current vector amplitude `amp` and are located outside the 'FOC/speed Controller' unit.
Moving one level into the `FOC/speed Controller-AFT` leads to the results shown in figure 4.243.
The `Current controller` and `Speed Controller` are as introduced earlier, see section 4.3.1.1. Sensorless FOC is realised with the Altair AFT®DLL, which has as inputs the per unit measured current vector and LPF voltage vector (in stationary α, β coordinates). Output is the `Cos_ang` and `Sin_ang` of the controller d-axis that must be aligned with the PM flux vector. The AFT module has a dialog box, as shown in figure 4.244 and the users must

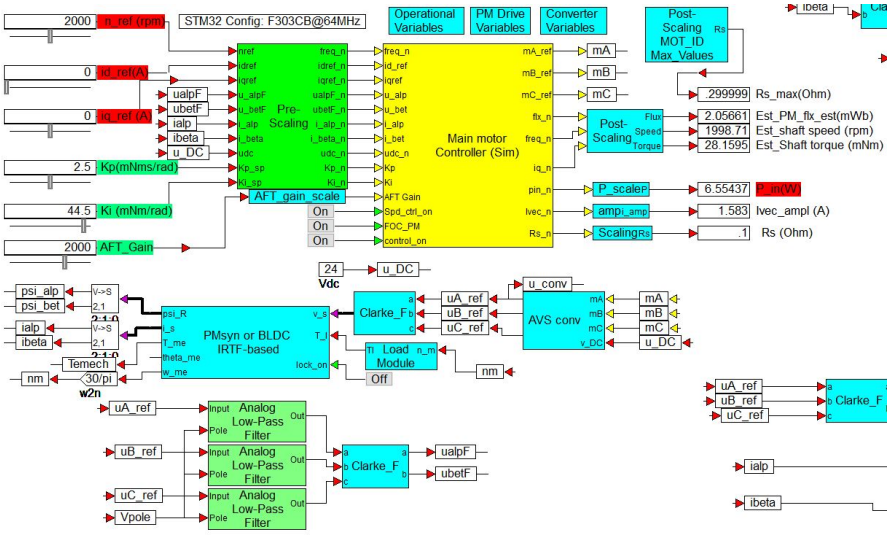


Fig. 4.241 Field-oriented torque/speed control using AFT®:SIM operation

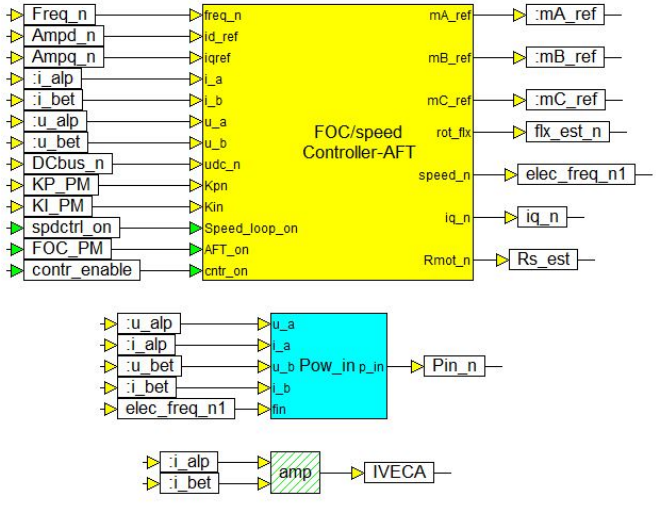


Fig. 4.242 Content of the 'main motor' module, field-oriented torque/speed control with AFT®

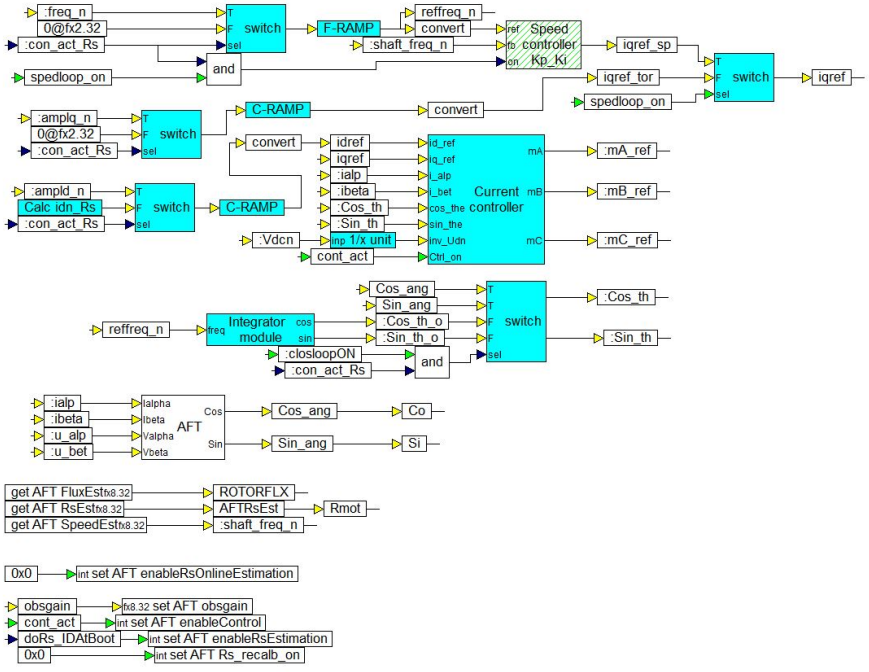


Fig. 4.243 Content 'FOC/Speed Controller-AFT' module, Field-oriented torque/speed control with AFT®

make sure that the content matches with the dialog box entries of modules to be discussed in the HIL subsection.

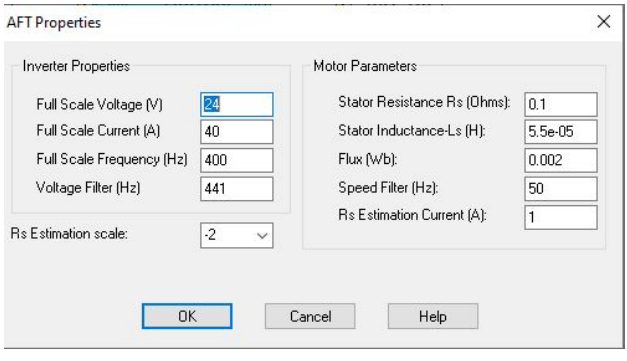


Fig. 4.244 Dialog box AFT module

This AFT module also provides an estimate of the per unit rotor flux `ROTORFLX`

and shaft speed `shaft_freq_n`. Torque is directly calculated using the rotor flux estimate together with the per unit measured quadrature current variable `iq` generated by the 'Current controller'. Also shown in figure 4.243 is variable `Rmot`, which estimates the stator resistance by injecting a direct axis current (amplitude set in the 'Operational Variables' dialogue box). When using a simulated model, as is the case here, this option is not used but is present in the HIL,shown in the next subsection.
When AFT is not used (encoderless off) the logic variable `closloopON` is zero and this implies open loop current&frequency control is active. However, the AFT®module will also remain active under these conditions, hence an effective mode to test operation of this module.

Running the SIM file by starting the file ('green arrow'), will start the simulation, which is set to operate with AFT active and speed control active (see figure 4.245).
Operation of the drive is at 2000 rpm and the load controls the quadrature current amplitude (once steady state operation is achieved), as mentioned earlier. AFT®operation is active and the torque is initially limited by the 5 A quadrature current setting for the speed controller.

The vector plot, which corresponds to time 1.0 s, is important, as it shows the scaled (by a factor of 750) rotor flux vector ('brown'), unity controller d-axis vector ('green') and terminal voltage vector ('blue').
The key observation from this vector plot is that the rotor flux vector and current vector (which is due the quadrature current) are orthogonal as required and expected. The current vector plot is trapezoidal due to the fact that a BLDC machine is present, which has a trapezoidal EMF. The latter also causes a slight error between the controller d-axis and the estimated flux vector, as can be observed from figure 4.246.

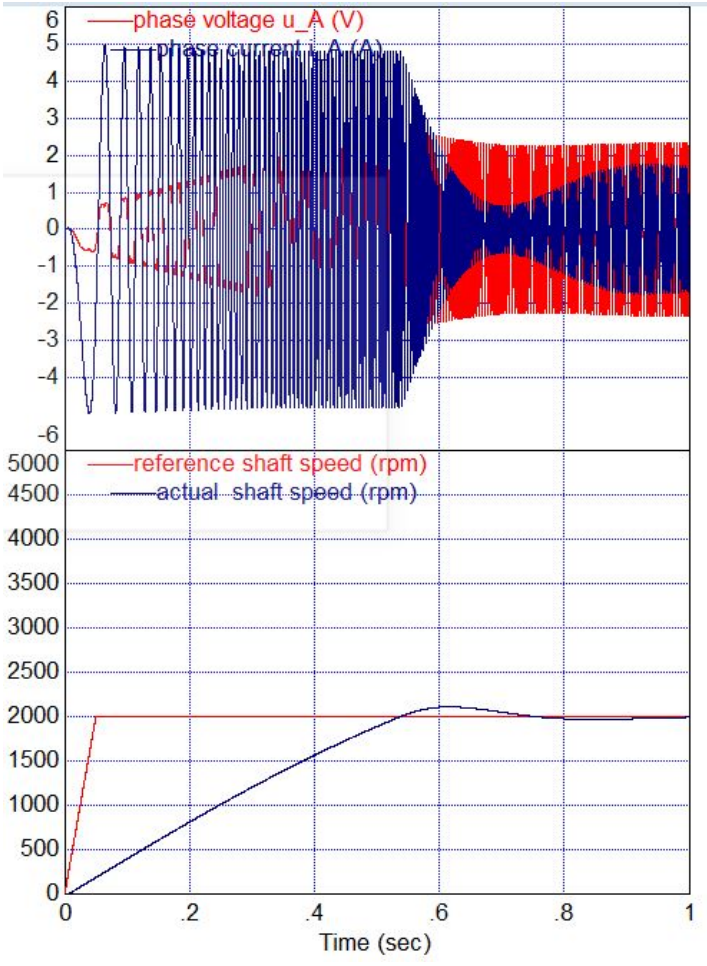


Fig. 4.245 SIM result: FOC sensorless operation using AFT®with speed control

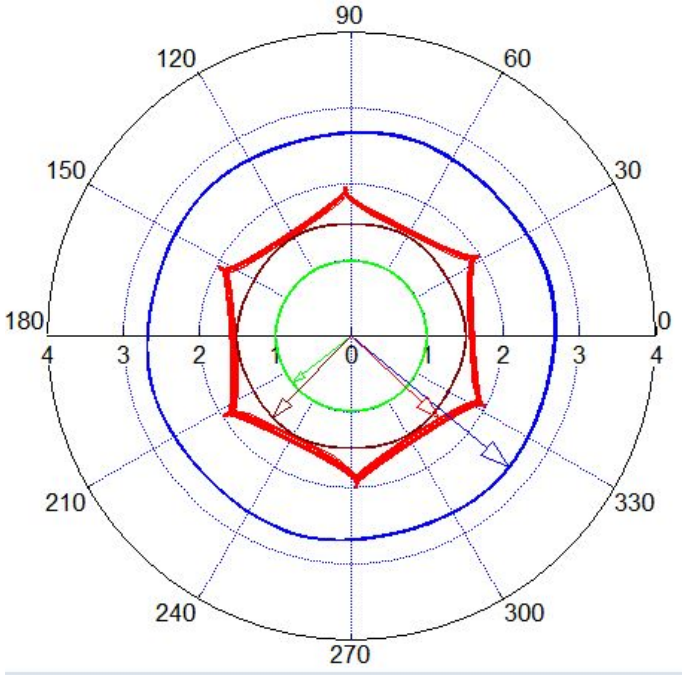


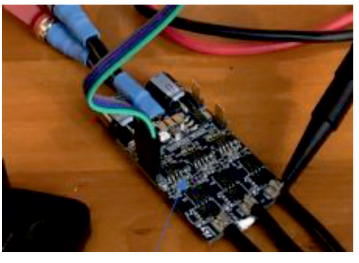
Fig. 4.246 SIM result, FOC sensorless operation using AFT®, with speed control: vector plot

4.6.1.2 AFT FOC-F303-ESC001:HIL

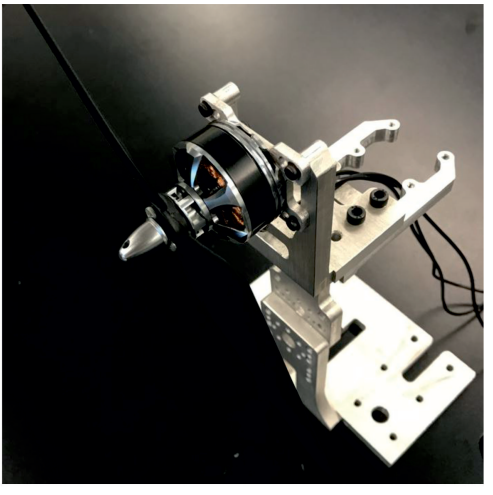
Reference files:

- `DLL-AFT_F303CB_ESC100_Drone_compile.vsm`, for HIL-compile
- `DLL-AFT_F303CB_ESC100_Drone_run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.247 represents the `EVAL ESC001` electronic speed controller (shown in figure 4.247(a)), which is connected to a BLDC drone motor[8] (shown in figure 4.247(b)). Said motor is mechanically connected to a 15x5R carbon composite propeller. The drone motor is mounted on an experimental platform developed by the department of electrical engineering, American University of Sharjah, UAE.



(a) STM ESC001



(b) Drone setup

Fig. 4.247 F303CB MCU-ESC001 setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.248 shows the `FOC-AFT and C&F Controller` module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes) for this

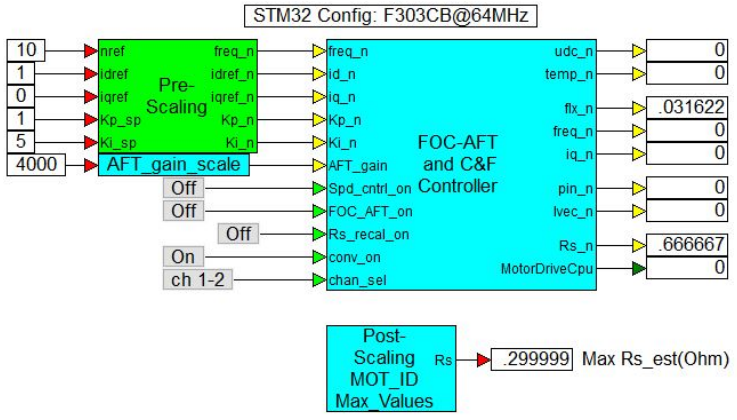


Fig. 4.248 HIL-Compile model of a BLDC drive with field-oriented sensorless control using AFT®

controller module are the user direct and quadrature axis reference currents `id`, `iq` and the reference shaft speed `nref`, which are converted to 'per unit' values in the 'Pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller. Also added to this example is an input for tuning the 'gain' used by AFT. This user setting is important for operation in the zero speed region, where the gain setting ensures adherence of the estimated flux to the flux reference set by the user.

Five logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed.
The other three are used as follows:

- `Sp_ctrl_on`: Speed loop control, when active the shaft speed slider controls the speed, based on the estimated shaft speed generated via AFT®.
- `FOC_AFT_on`: Activates sensorless FOC using AFT®. Note when OFF the AFT®algorithm is still operational. This is useful when C&F control is active (both `Sp_loop_on` and `FOC_AFT_on` buttons: OFF).
- `Rs_recal_on`: Measures the stator resistance at start-up (when `conv_on` is ON) by injection of a direct axis current (value set in the 'Operational Variables' dialogue box). The duration of the stator resistance estimation time is set in the `100Hz` module via a time delay setting. It is important to verify that the estimation time is sufficient to reach a steady state resistance estimate.
- `conv_on`: Activates the controller.
- `chan_sel`: Determines which two measurement signals are shown on the plot unit.

Outputs of the Controller module are:

- `udc_n`: The per unit DC bus voltage, which must be scaled by the post-scaling module (in the HIL-debug phase) to provide a reading in Volts.
- `temp_n`: The converter temperature reading, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in degrees C.
- `flx_n`: The scaled rotor flux estimate generated by AFT®.
- `freq_n`: A per unit shaft speed estimate `frq_n` generated via AFT®.
- `iq_n`: The measured per unit quadrature current, used to generate a shaft torque estimate by making use of the estimated flux and pole pair number.
- `pin_n`: A per unit estimate of the motor input power. This variable is also useful as an estimate for the drive input power (but this ignores the converter losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx {}^{p_{\mathrm{in}}}/_{u_{\mathrm{DC}}}$.
- `Ivec_n`: The per unit current vector amplitude.
- `Rs_n`: The per unit stator resistance value generated by the resistance estimator at start-up (when `Rs_recal_on` is ON). When selected, the estimated stator resistance value is transferred to the AFT algorithm, on completion of the estimation sequence.
- `CPU_activity`: Provides the user with a % reading of MCU activity, relative to the controller sampling time.

Also shown in figure 4.248 is a `Post Scaling Motor ID Max_Values` module and its value shows the maximum resistance value that can be used by AFT. This value is set by the user via a dialog box entry, to be discussed at a later stage.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 4.249.

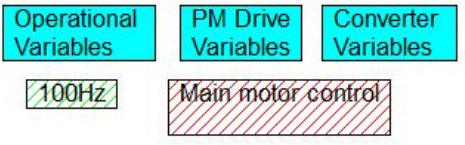


Fig. 4.249 One level down into the FOC-AFT module

Three dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. Note that the AFT dialog box values (see figure 4.244) must match with the subset of variables given in the boxes shown in figure 4.249. The 'Operational Variables' dialogue box

(see figure 4.250), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current(s)-frequency. In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the speed controller sampling frequency is typically set lower then used by the main controller. In this case it is set to 1 kHz. These settings are similar to those introduced for an FOC drive with encoder, as discussed in section 4.3.1.4 (for a different MCU). This module also shows the`Speed filter pole`: corner frequency of the first order LP filter used by the AFT®speed controller.
The variables:

- `F_est/F_sampling`: sets the sampling rate used by AFT®relative to the controller sampling rate, typically set to 1.
- `F_speed/F_sampling`: sets the sampling rate of the speed controller relative to the sample rate of the main controller.
- `PWMFreq(Hz)`: is the PWM frequency used by the converter.
- `Fsampling(Hz)`: is the sampling frequency used by the main controller module.

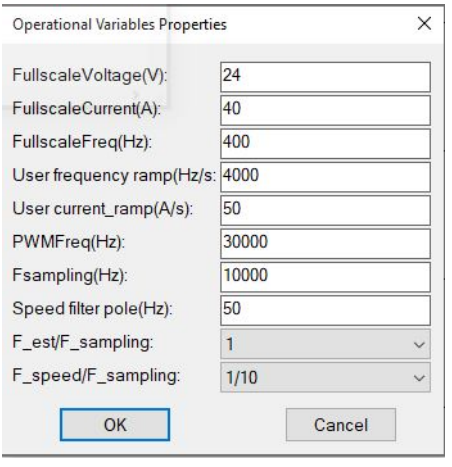


Fig. 4.250 HIL: Dialogue box 'Operational Variables'

The 'Converter Variables' dialogue box (shown in figure 4.249), defines the scaling used by the Analogue to Digital converter, as discussed earlier, see figure 4.103.
This module also shows the low-pass filter corner frequency used by the converter for measuring the half-bridge voltages (see figure 4.95) and is an important parameter for AFT®, as it is used to compensate for speed dependant phase angle and amplitude errors of the LPF terminal voltage vector.

The 'PM Drive Variables' dialogue box (see figure 4.251), shows the current controller gain/bandwidth and maximum modulation index value for the drive. Furthermore, the motor pole pair value and speed controller current limit `speed_lim (A)` (which is the maximum quadrature current setting) are defined in this dialog box.

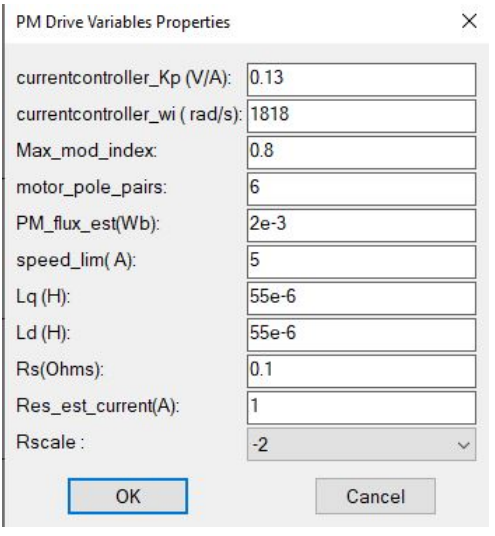


Fig. 4.251 HIL: Dialogue box 'PM Drive Variables'

Also added to this dialogue module are a number of AFT®and PM machine parameters namely:

- `PM_flux)est (Wb)`: an important parameter which represents an estimate for the PM rotor flux in Wb. This value can be estimated, by using the drive in the C&F mode, as shown earlier for a different machine, see the 'Post Analysis' 4.2.1.5 example. Typically the value entered here, should be equal to the PM flux value of the machine in use.
- `Rs (Ohms)`: the stator resistance value which is an important AFT®variable at startup of the drive. In this context the EMF threshold speed has been introduced (see equation (2.12a) which for this drive is equal to 398 rpm, which shows the importance of accurate stator resistance values.
- `Lq (H)`: the quadrature axis inductance, which for this non-salient machine is equal to the stator inductance and can be measured using an L-C-R meter (as required to set the current controller gain and bandwidth).
- `Ld(H)`: The direct axis inductance, which for PMsyn and BLDC is equal to the quadrature inductance.
- `p`: Pole pair number, which can be found by operating the current in C&F mode and observing the rotational speed of the rotor when, for example

setting the reference shaft speed to 60 rpm (one rotation per second). Alternatively, for 'outrunner' BLDC machines the number of rotor magnets can be counted and then halved to yield the pole pair number.
- `Res_est_current(A)`: the direct axis current amplitude used to measure the stator resistance at start-up, when the option `Rs_recal_on`, is selected by the user.
- `Rscale`: This variable is used to set the maximum stator resistance value of the module `Post Scaling MOT_ID Max values` (see figure 4.248). This value is based on measurement of the stator resistance by (for example) using an L-C-R meter and adding a 30% margin.

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently be displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active. Furthermore, this unit generates a number of AFT®settings, which can generally be left unchanged.

The key component shown in figure 4.249 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 4.252. Of these, the 'FOC/speed Controller-AFT' has been discussed earlier (see figure 4.243), where it was used in phase: SIM of the development tool-chain.

The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 30 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0.2 μs
- Board in use: `ESC001` with `F303CB` MCU

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after every third (for this drive) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

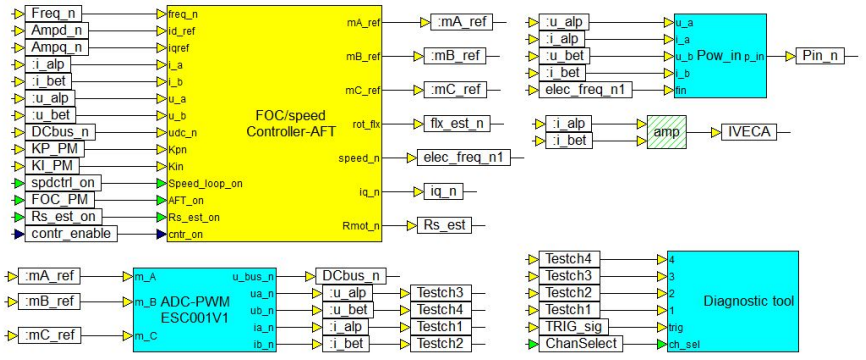


Fig. 4.252 One level down into the 'Main motor controller' module

The `ADC-PWM-ESC001V1` unit is used generate the PWM signals for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the `100Hz` module) and controlled by said module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 4.253 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. Three sliders are used to set the direct/quadrature axis current reference values and the electrical frequency. Two additional sliders are used to set the proportional and integral gain of the speed controller.
Post-scaling modules are used to convert the per-unit measured DC bus voltage, temperature and % CPU time to actual values, as shown with numeric displays. In addition, a set of numerical displays is present which show: estimated PM flux, estimated shaft speed, input power, current vector amplitude, measured stator resistance at startup (if this function is activated) and estimated shaft torque. Furthermore, a set of gauges are present which monitor (low-pass filtered) variables: shaft torque, shaft speed, input power and DC supply current.
Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.
The plot given in figure 4.253, displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 1-2`. Selecting

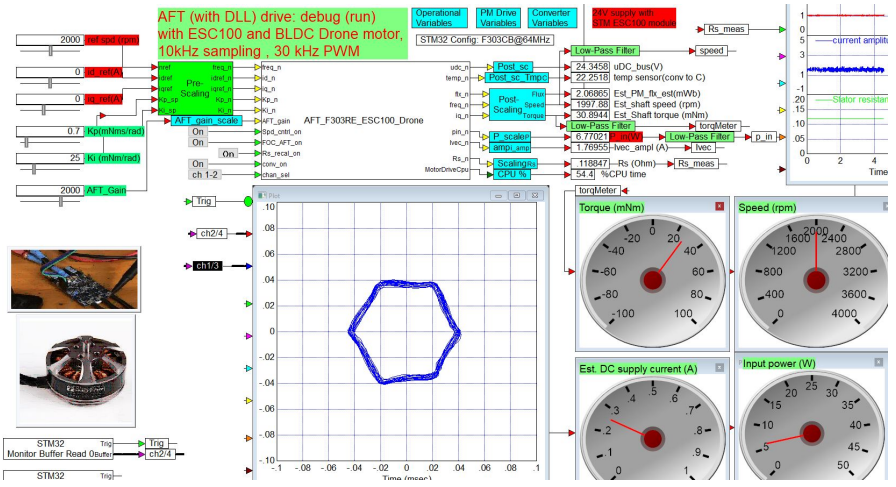


Fig. 4.253 HIL-Debug model, F303-ESC001 of a BLDC drive with field-oriented control using AFT®

`ch 3-4`, will display the per unit LPF voltage vector locus. The electrical shaft speed reference in figure 4.253 is set to 2000 rpm. These results shows operation under encoderless FOC-speed control with the machine operating under load conditions.

For comparison purposes, the phase current is shown in figure 4.254, which was measured with this aid of a 'Chauvin Arnoux' E27 AC/DC current clamp E27, which is connected to a 'PicoScope 6' unit that provides a 2 channel oscilloscope [20] capability. These results clearly show the presence of a sixth harmonic in the current and the PWM switching effects for the drive operating at 2000 rpm.
For this application an indication of FOC performance limit can be found using the Embed file shown in figure 4.297. However, in this case the maximum quadrature current used by the speed controller must be set to 5.0 A, which implies that the maximum speed (for FOC control) is equal to 7490 rpm for this application.

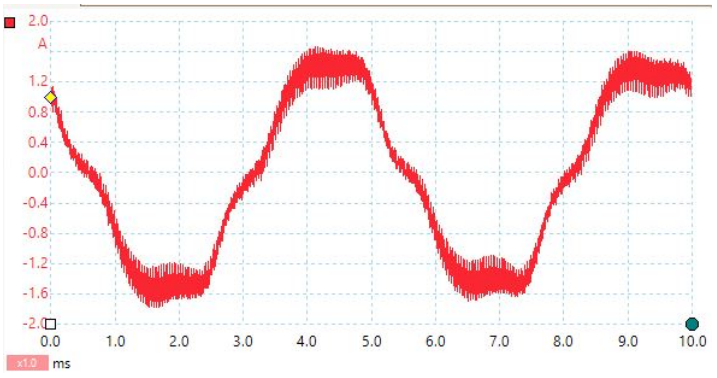


Fig. 4.254 Measured phase current, F303-ESC001 of a BLDC drive with field-oriented control using AFT®

Post Analysis

Present in the debug file is a time plot that shows the estimated rotor flux amp normalised to the rotor flux value set in the 'PM Drive Variables' dialogue box. If the value set in said dialog box, matches the measured value then the normalised flux shown in figure 4.255 will be 1, as is indeed the case.

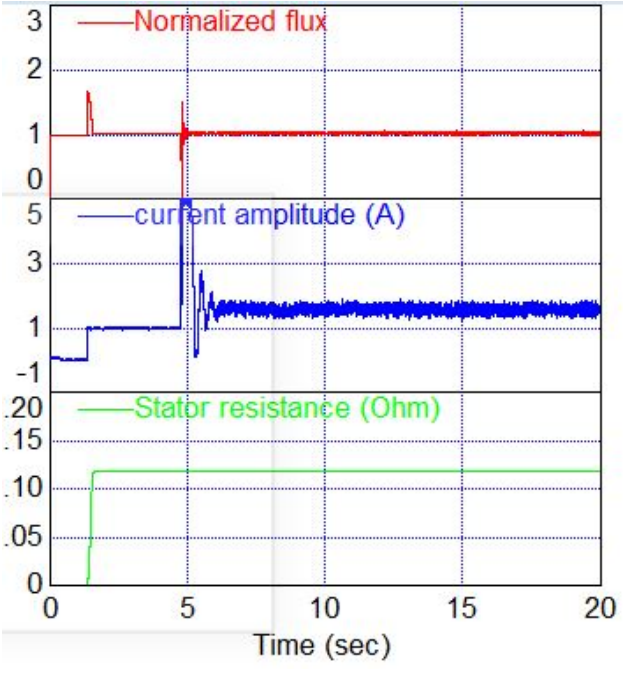


Fig. 4.255 Measured data, F303-ESC001 of a BLDC drive with field-oriented control using AFT®

The results shown in figure 4.255, were undertaken at start-up and then operation was terminated after 20 s. The reason for this was to show stator resistance estimation, which shows that a 1 A current was used (as is apparent from the vector current amplitude sub-plot, also shown). After resistance estimation, drive operation starts and the 5.0 A speed controller limit is initially reached and the current settles to its steady state value, dictated by the mechanical load (the propeller in this case).

4.6.2 Sensorless Field Oriented Control using AFT®-TI

The examples linked to Texas Instruments will be covered using the development toolchain components: SIM→HIL, based on the use of the F280039. This MCU wil be used to run 'Altair Flux Tracker' (AFT), which is a classified sensorless algorithm AFT®and access to said algorithm is by way of an Embed DLL. Initially the tool-chain components presented in the ensuing subsections will be based on the use of the F280039-GANINV drive setup with a BLDC drone [8].

4.6.2.1 AFT FOC-F0039-GANINV:SIM

Reference file:

- `DLL-AFT_F280039-GANINV-Sim.vsm`

The simulation (SIM) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F280039, which contains the DLL used to communicate with AFT®(see figure 4.256).

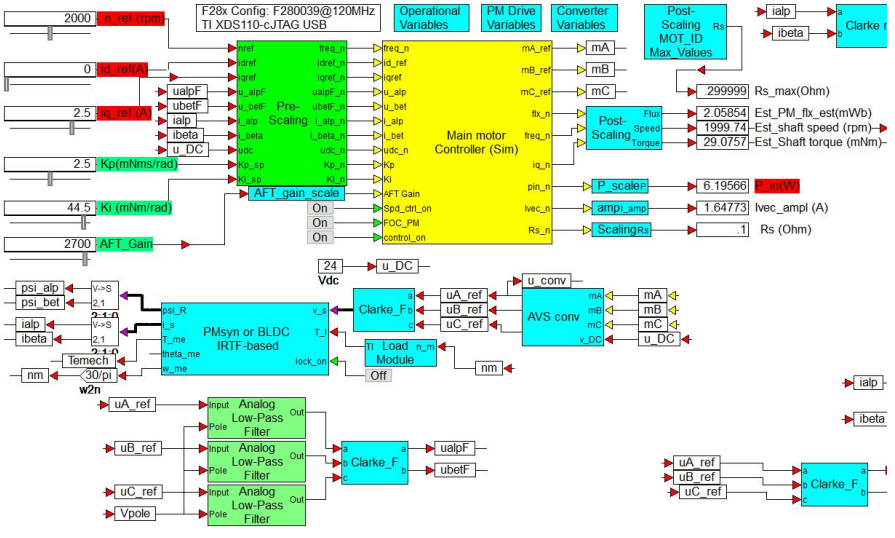


Fig. 4.256 Field-oriented torque/speed control using AFT®:SIM operation

The module `Main motor Controller(Sim)`, generates the modulation indices required for the converter/motor. In this example the IRTF machine model is

set to operate in BLDC mode (not PMsyn, given the use of a BLDC machine). The three LPF modules, are used to model the converter R-C network used to measure the motor voltage vector. Moving one level down the 'Main motor controller(Sim)' leads to the contents shown in figure 4.257.

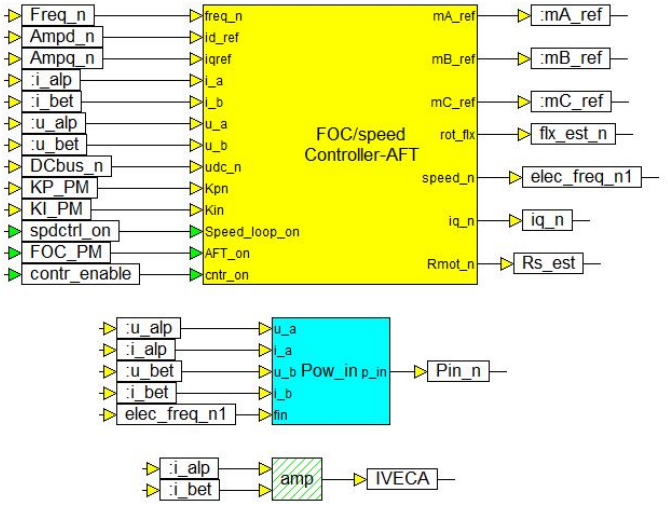


Fig. 4.257 Content 'main motor' module, Field-oriented torque/speed control with AFT®

Shown in said figure are the `FOC/speed Controller-AFT` module and two additional modules which measure input power (with per unit output `Pin_n`) and current vector amplitude (with per unit output `IvecA`). Moving one level into the `FOC/speed Controller-AFT` leads to the content shown in figure 4.258.
The `Current controller` and `Speed Controller` are as introduced earlier, see section 4.3.1.1. Sensorless FOC is realised with the Altair AFT®DLL, which has as inputs the per unit measured current vector and LPF voltage vector (in stationary α, β coordinates). Output is the `Cos_ang` and `Sin_ang` of the controller d-axis that must be aligned with the PM flux vector. The AFT module has a dialog box, as shown in figure 4.259 and the users must make sure that the content matches with the dialog box entries of modules to be discussed in the HIL subsection.

This AFT module also provides an estimate for the per unit rotor flux `ROTORFLX` and shaft speed `shaft_freq_n`. Torque is directly calculated using the rotor flux estimate together with the per unit measured quadrature current variable `iq` generated by the 'Current controller'. Also shown in figure 4.258 is variable `Rmot`, which estimates the stator resistance by injecting a direct axis

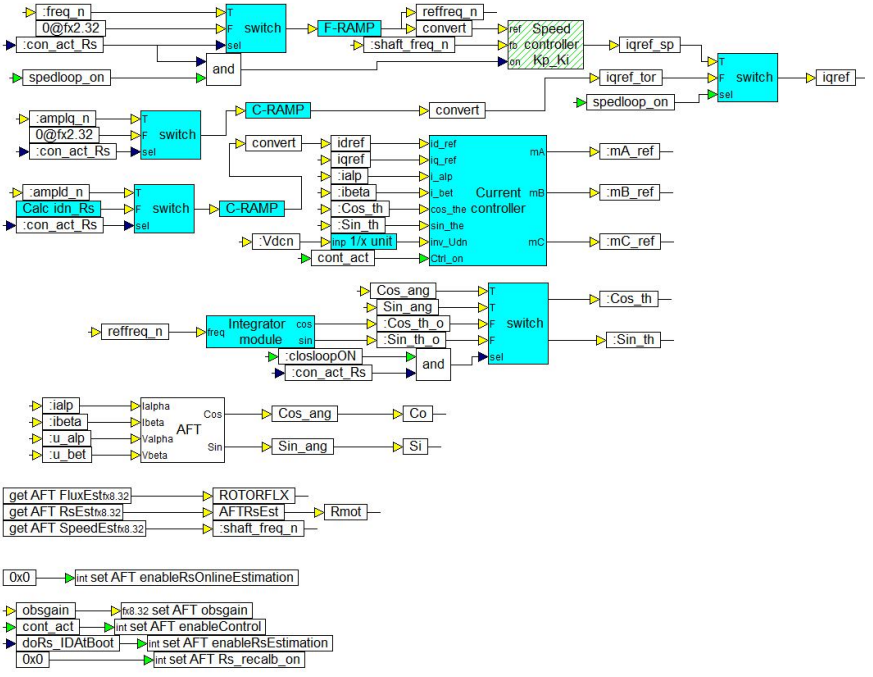


Fig. 4.258 Content 'FOC/Speed Controller-AFT' module, Field-oriented torque/speed control with AFT®

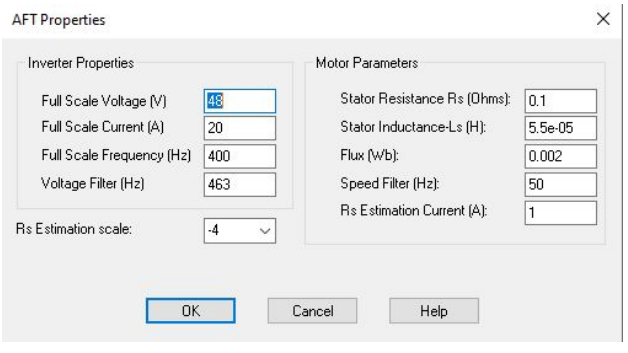


Fig. 4.259 Dialog box AFT module

current (amplitude set in the 'Operational Variables' dialogue box). When using a simulated model, as is the case here, this option is not used, but is present in the HIL shown in the next subsection.
When AFT is not used (encoderless off) the logic variable `closloopON` is zero and this implies open loop current&frequency control is active. However, the AFT®module will also remain active under these conditions, hence an effective mode to test operation of this module.

Running the SIM file by starting the file ('green arrow') will start the simulation, which is set to operate with AFT active and speed control active (see figure 4.260).

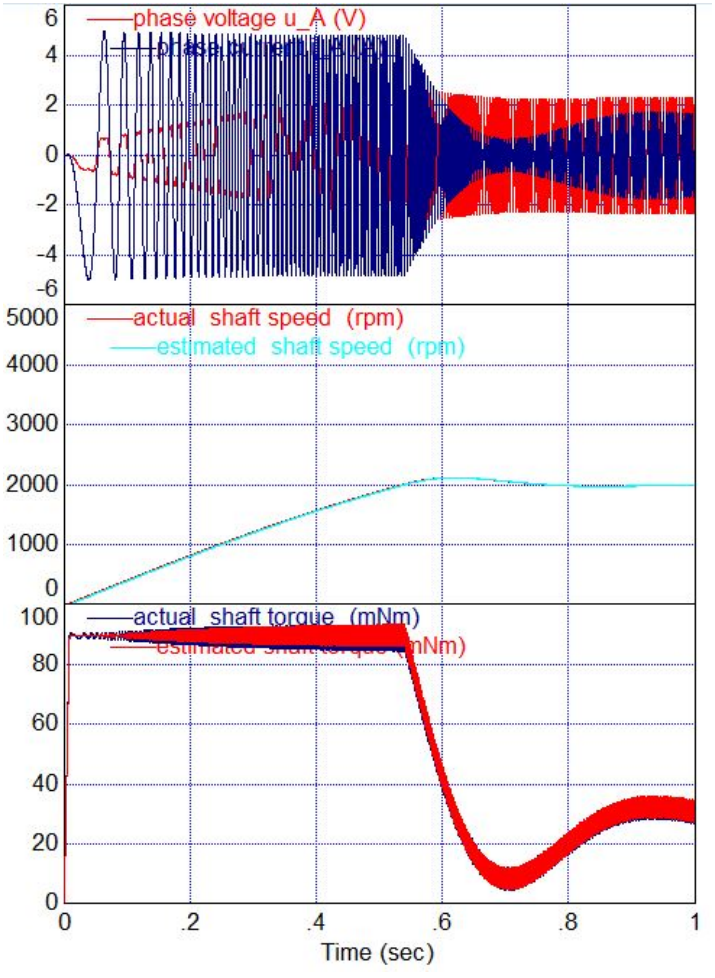


Fig. 4.260 SIM Result: FOC sensorless operation using AFT®with speed control

Operation of the drive is at 2000 rpm and the load controls the quadrature current amplitude (once steady state operation is achieved), as mentioned earlier. AFT®operation is active and the torque is initially limited by the 5 A quadrature current setting for the speed controller. Also shown in said figure are the estimated torque and speed, generated by the controller.

The vector plot, which corresponds to time mark 1.0 s, is important as it shows the scaled (by a factor 750) rotor flux vector ('green'), current vector ('red') and scaled (by a factor 1/3)terminal voltage vector ('blue').

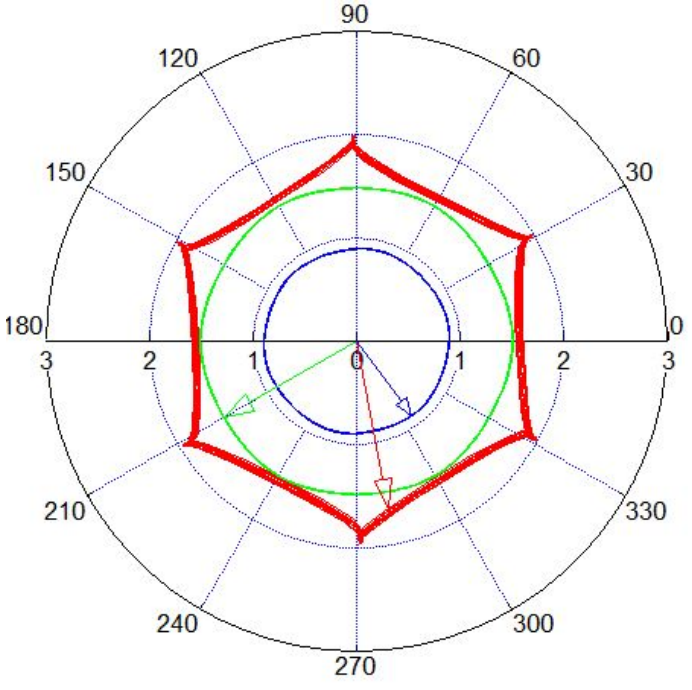


Fig. 4.261 SIM result, FOC sensorless operation using AFT®with speed control: vector plot

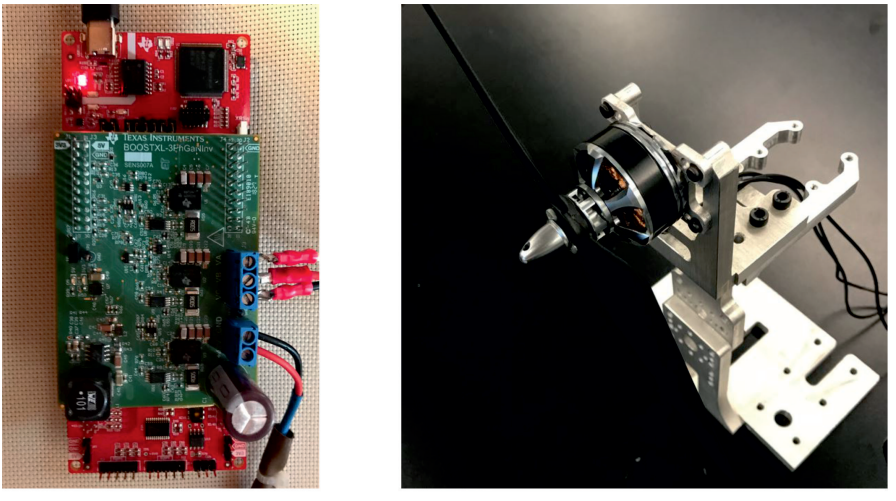
The key observation from this vector plot is that the rotor flux vector and current vector (which is due the quadrature current) are approximately (because a BLDC is used here) orthogonal. The current vector plot is trapezoidal due to the fact that a BLDC machine is present which has a trapezoidal EMF.

4.6.2.2 AFT FOC-F0039-GANINV:HIL

Reference files:

- `DLL-AFT-F280039-GANIN-compile.vsm`, for HIL-compile
- `DLL-AFT-F280039-GANIN-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.262 represents the `F280039-GANINV` converter-launchpad (see figure 4.262(a)), which is connected to a BLDC drone motor[8] (shown in figure 4.262(b)). Said motor is mechanically connected to a 15x5R carbon composite propeller. The drone motor is mounted on an experimental platform developed by the department of electrical engineering, American University of Sharjah, UAE.



(a) F280039-GANINV (b) Drone setup

Fig. 4.262 F280039 MCU-GANINV setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches.
A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.263 shows the `FOC-AFT and C&F Controller` module, which must be compiled to generate an out-file.

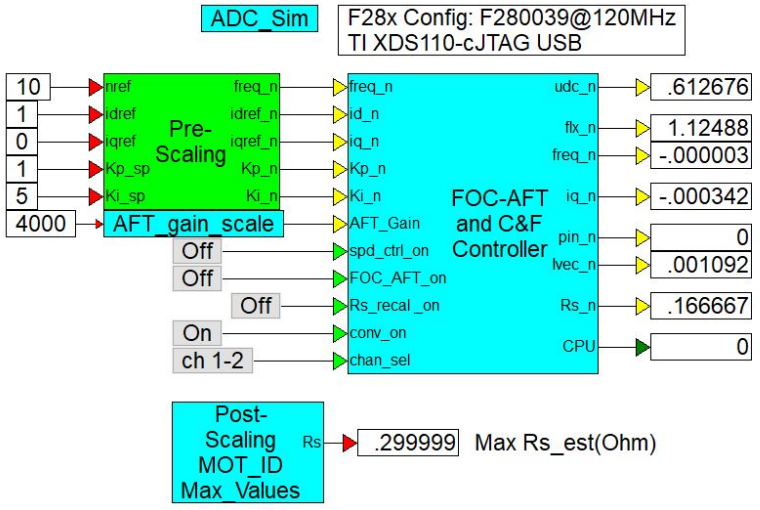


Fig. 4.263 HIL-Compile model of a BLDC drive with field-oriented sensorless control using AFT®

Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct and quadrature axis reference currents `id`, `iq` and the reference shaft speed `nref`, which are converted to 'per unit' values in the 'Pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller. Also added to this example is an input for tuning the 'gain' used by AFT. This user setting is important for operation in the zero speed region where the gain setting ensures adherence if the estimated flux to the flux reference set by the user.
Five logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other three are used as follows:

- `Sp_ctrl_on`: Speed loop control, when active the shaft speed slider controls the speed, based on the estimated shaft speed generated via AFT®.
- `FOC_AFT_on`: Activates sensorless FOC using AFT®. Note when OFF the AFT®algorithm is still operational and this is useful when C&F control is active (both `Sp_loop_on` and `FOC_AFT_on` buttons:OFF).
- `Rs_recal_on`: Measures the stator resistance at start-up (when `conv_on` is ON) by injection of a direct axis current (value set in the 'operational variables' dialogue box). The duration of the stator resistance estimation time is set in the `100Hz` module via a time delay setting. It is important to verify that the estimation time is sufficient to reach a steady state resistance estimate.
- `conv_on`: Activates the controller.

- `chan_sel`: Determines which two measurements are shown on the scope unit.

Outputs of the Controller module are:

- `udc_n`: The per unit DC bus voltage, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- `flx_n`: The scaled rotor flux estimate generated by AFT®.
- `freq_n`: A per unit shaft speed estimate `frq_n` generated via AFT®.
- `iq_n`: The measured per unit quadrature current, used to generate a shaft torque estimate by making use of the estimated flux and pole pair number.
- `pin_n`: A per unit estimate of the motor input power. This variable is also useful as an estimate for the drive input power (bus this ignores the converter losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx {}^{p_{\mathrm{in}}}/_{u_{\mathrm{DC}}}$.
- `Ivec_n`: The per unit current vector amplitude.
- `Rs_n`: The per unit stator resistance value generated by the resistance estimator at start-up (when `Rs_recal_on` is ON). When selected the estimated stator resistance value is transferred to the AFT algorithm on completion of the estimation sequence.
- `CPU`: Provides the user with a % reading of MCU activity, relative to the controller sampling time.

Also shown in figure 4.263 is a `Post Scaling Motor ID Max_Values` module and its value shows the maximum resistance value that can be used by AFT. This value is set by the user via a dialog box entry, to be discussed at a later stage.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 4.264.

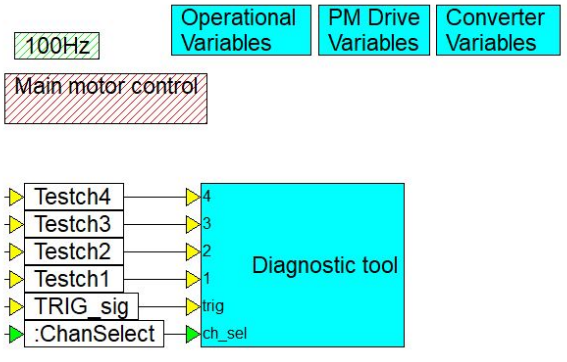


Fig. 4.264 One level down into the FOC-AFT module

Three dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. Note that the AFT dialog box values (see figure 4.259) must match with the subset of variables given in the boxes shown in figure 4.264. The 'Operational Variables' dialogue box (see figure 4.265), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current(s)-frequency. In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the speed controller sampling frequency is typically set lower then used by the main controller. In this case it is set to 1 kHz. These settings are similar to those introduced for an FOC drive with encoder, as discussed in section 4.3.1.4. This module also shows the `Speed filter pole`: corner frequency of the first order LP filter used by the AFT®speed controller.
The remaining variables shown in this dialog box are:

- `F_est/F_sampling`: sets the sampling rate used by AFT®relative to the controller sampling rate, typically set to 1.
- `F_speed/F_sampling`: sets the sampling rate of the speed controller relative to the sample rate of the main controller.
- `PWMFreq(Hz)`: is the PWM frequency used by the converter.
- `Fsampling(Hz)`: is the sampling frequency used by the main controller module.

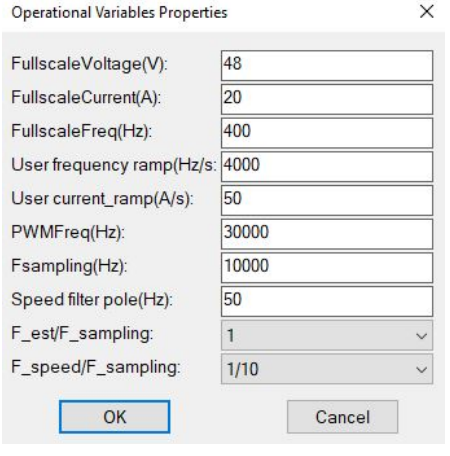


Fig. 4.265 HIL: Dialogue box 'Operational Variables'

The 'Converter Variables' dialogue box, defines the scaling used by the Analogue to Digital converter, as discussed earlier (see figure 4.191).
Said module also shows the low-pass filter corner frequency used by the

converter for measuring the half-bridge voltages (see figure 4.12) and is an important parameter for AFT®, as it is used to compensate for speed dependant phase angle and amplitude errors of the LPF terminal voltage vector.

The 'PM Drive Variables' dialogue box, as given in figure 4.266, shows the current controller gain/bandwidth and maximum modulation index value for the drive. Furthermore, the motor pole pair value and speed controller current limit `speed_lim (A)` (which is the maximum quadrature current setting), are defined in this dialog box.

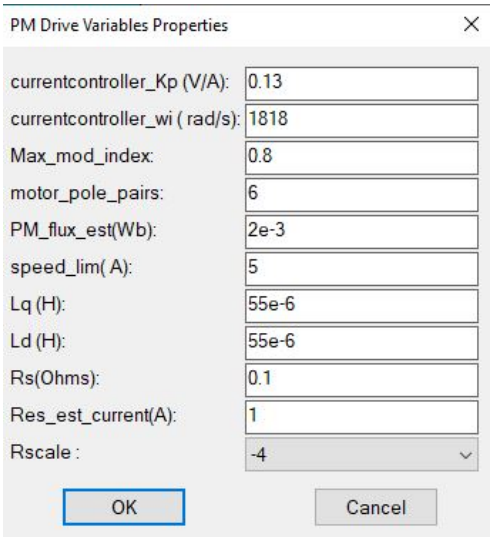


Fig. 4.266 HIL: Dialogue box 'PM Drive Variables'

Also added to this dialogue module are a number of AFT®and PM machine parameters namely:

- `PM_flux)est (Wb)`: an important parameter which represents an estimate for the PM rotor flux in Wb. This value can be estimated, by using the drive in the C&F mode, as shown earlier for a different machine, see the 'Post Analysis' 4.2.1.5 example . Typically the value entered here, should be equal to the PM flux value of the machine in use.
- `Rs (Ohms)`: the stator resistance value which is an important AFT®variable at startup of the drive. In this context the EMF threshold speed has been introduced (see equation (2.12a) which for this drive is equal to 398 rpm, which shows the importance of accurate stator resistance values.
- `Lq (H)`: the quadrature axis inductance, which for this non-salient machine is equal to the stator inductance and can be measured using an L-C-R meter (as required to set the current controller gain and bandwidth).

- `Ld(H)`: The direct axis inductance, which for PMsyn and BLDC is equal to the quadrature axis inductance.
- `p`: Pole pair number, which can be found by operating the current in C&F mode and observing the rotational speed of the rotor when, for example setting the reference shaft speed to 60 rpm (one rotation per second). Alternatively for 'outrunner' BLDC machines the number of rotor magnets can be counted and then halved to yield the pole pair number.
- `Res_est_current(A)` : the direct axis current amplitude used to measure the stator resistance at start-up when the option `Rs_recal_on` is selected by the user.
- `Rscale`: This variable is used to set the maximum stator resistance value of the module `Post Scaling MOT_ID Max values` (see figure 4.263). This value is based on measurement of the stator resistance by (for example) using an L-C-R meter and adding a 30% margin.

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active. Furthermore, this unit generates a number of AFT®settings, which can generally be left unchanged.

The key component shown in figure 4.264 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit.
Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 4.267. Of these, the 'FOC/speed Controller-AFT' has been discussed earlier (see figure 4.258), where it was used in phase: SIM of the development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module.
This module is configured with the following basic settings :

- PWM frequency: 30 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F280039` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because the gate drives are present that handle this issue. This ADC-PWM module conforms with MCU timing, as

discussed in figure 2.40, which implies that the `Main motor control` module is executed after every third (for this drive) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

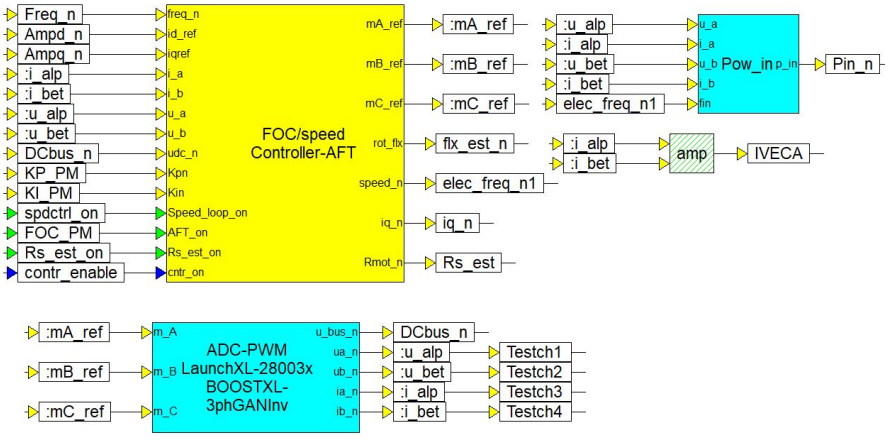


Fig. 4.267 One level down into the 'Main motor controller' module

The `ADC-PWM-LaunchXL-28003x` unit is used generate the PWM signals for the converter. In addition it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively.
ADC offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the `100Hz` module) and controlled by said module, as mentioned previously. The per unit DC bus voltage `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example (see figure 4.268), and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. Three sliders are used to set the direct axis/quadrature current reference values and the electrical frequency. In addition, two sliders are present for setting the proportional and integral gain of the speed controller. Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Furthermore, a set of numerical display units are present which show: estimated PM flux, estimated shaft speed, estimated shaft torque, input power, current vector amplitude and estimated stator resistance at start-up (if this

function is selected). Four gauges are present, which monitor (low-pass filtered): shaft torque, shaft speed, input power and DC supply current.
Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

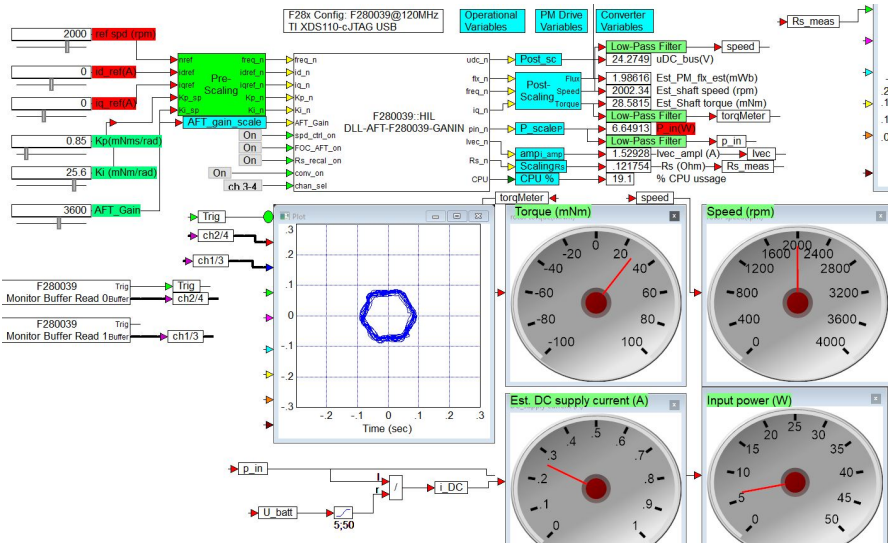


Fig. 4.268 HIL-Debug model, F280039-GANINV based BLDC drive with field-oriented control using AFT®

The plot given in figure 4.268 displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit LPF voltage vector locus. The electrical shaft speed reference in figure 4.268 is set to 2000 rpm. These results shows operation under encoderless FOC-speed control with the machine operating under load conditions.

For comparison purposes, the phase current is shown in figure 4.269, which was measured with this aid of a 'Chauvin Arnoux' E27 AC/DC current clamp E27, which is connected to a 'PicoScope 7' unit that provides a 2 channel Oscilloscope [20] capability. These results clearly show the presence of a sixth harmonic in the current and the PWM switching effects for the drive operating at 2000 rpm. For this application an indication of FOC performance limit can be found using the Embed file shown in figure 4.297. However, in this case the maximum quadrature current used by the speed controller must be set to 5.0 A, which implies that the maximum speed (for FOC control) is equal to 7177 rpm for this application (assuming zero direct axis current).

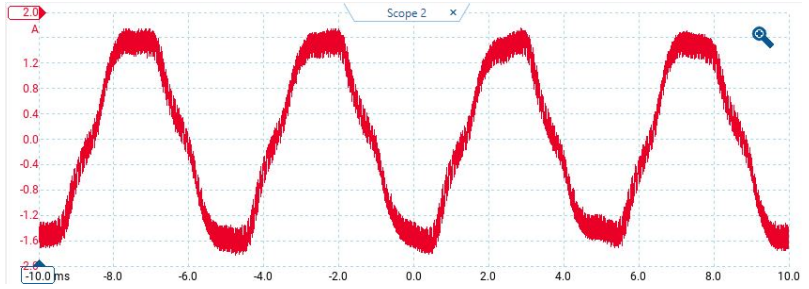


Fig. 4.269 Measured phase current, F280039-GANINV based BLDC drive with field-oriented control using AFT®

4.6.3 Sensorless Field Oriented Control using FAST®-TI

The examples linked to Texas Instruments will be covered using the development toolchain components: PIL→COSIM→HIL, based on the use of the Piccolo [23] series: F28027 and F28069 Launchpad's. These MCU's have access to the Texas Instruments software InstaSPIN®, which contains their classified sensorless algorithm FAST®. Access to FAST is by way of an Embed DLL, which means that simulation is not an option when using this algorithm. Initially the tool-chain components presented in the following subsections will be based on the use of the F28069-GANINV drive setup with a PMsyn machine [2], but other HIL examples will also be given.

4.6.3.1 FAST FOC-F069-GANINV:PIL

Reference files:

- `FOC_PM_FAST_F069-GANINV-PIL.vsm`
- `FOC_FAST-PIL_mod.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F28069, which contains the DLL used to communicate with FAST. This requires the use of a target execution module for file, `FOC_FAST-PIL.out`, as indicated in figure 4.270.
The target interface module that executes the out-file, generates the modulation indices required for the converter/motor and these components are not affected by use of the PIL. In this example the IRTF machine model is set to operate in PMsyn mode (not BLDC, given the use of a PM synchronous machine). Generation of the out file is based on the PIL module, see figure 4.271, which must be compiled using the command: `toolbox>codegen`.

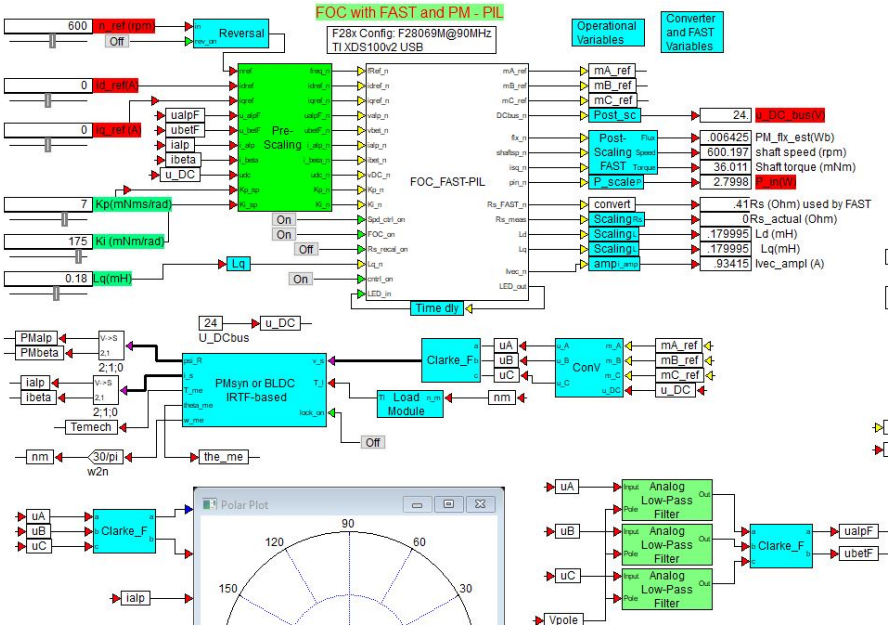


Fig. 4.270 Field-oriented torque/speed control using FAST®: PIL operation

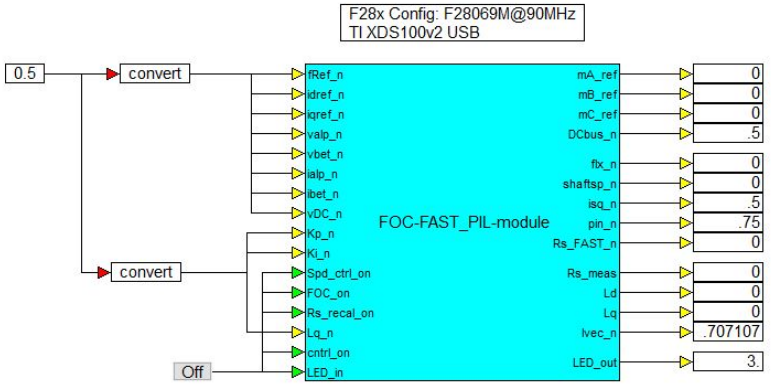


Fig. 4.271 Module:'FOC-FAST-PIL'

Moving one level into the PIL module `FOC-FAST-PIL`, shows a 'Drive Controller' (see figure 4.272).
Added to said controller is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that the PIL control is operational. Furthermore, the modules responsible for input power measurement `Pow_in` and current vector amplitude `amp` are located outside the 'Drive Controller'.

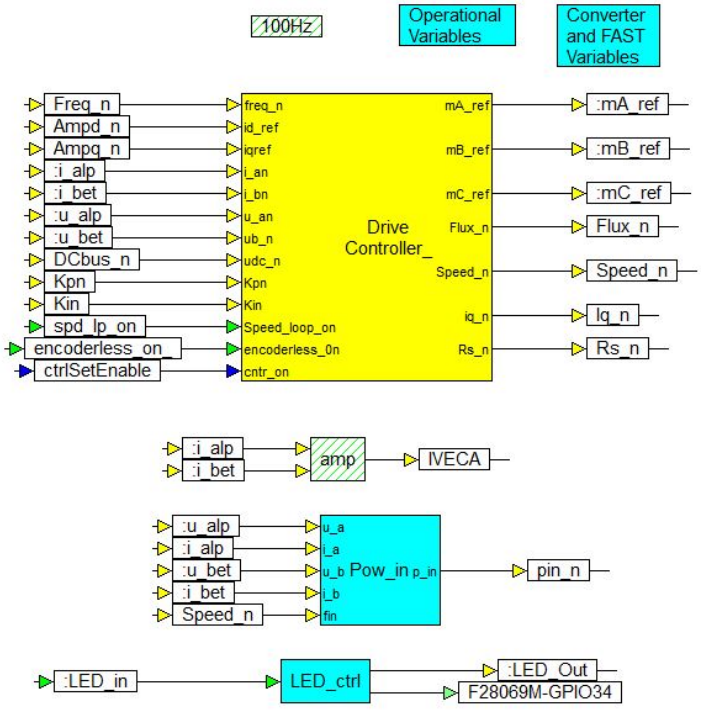


Fig. 4.272 Content PIL module, Field-oriented torque/speed control with FAST®

Drilling into the `FOC-FAST_PIL-module`, leads to the content shown in figure 4.273. The `Current controller` and `Speed Controller` are as introduced earlier, see section 4.3.1.1.

However, the per unit inverse bus voltage variable `invVdc`, is now generated by the FAST®module `InstaSPIN FOC`. Said unit provides per unit estimates of the rotorflux amplitude `ROTORFLX`, rotor flux angle `ang`, and electrical frequency `rotfreq`. This module can also provide an estimate for the torque `eleTT`, but that is not used here, as it is directly calculated using the rotor flux estimate together with the per unit measured quadrature current variable `iq` generated by the 'Current controller'.
Inputs to the FAST®module are the per unit measured LPF voltages and currents, which are Clarke transformed to stationary α, β coordinates. In addition, a variable `CurCTLstate` (generated in the `100 Hz` module) activates the FAST®module, which is also provided with the per unit measured DC bus voltage `Vdcn`. Also shown in figure 4.273 is a module `Rs_estimation`, which (when activated by the user with variable `Rs_ID`), estimates the stator resistance by injecting a direct axis current (amplitude set in the 'Operational

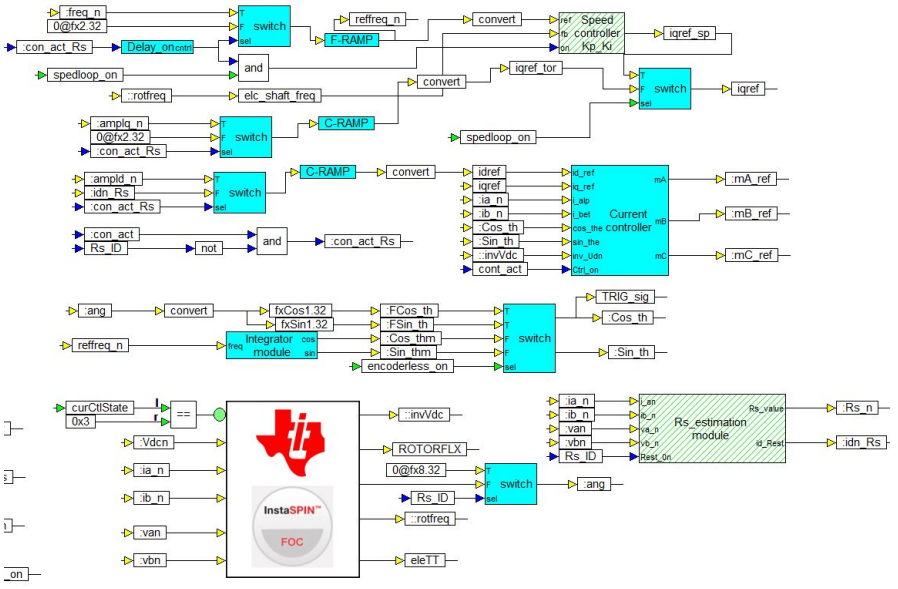


Fig. 4.273 Content 'Drive Controller' module

Variables' dialogue box). When stator estimation is active the angle output of the FAST®module is held to zero.
When FAST is not used (encoderless off) the logic variable `encoderless_on` is zero and this implies open loop current&frequency control is active. However, the FAST®module will remain active under these conditions, hence an effective mode to test operation of this module.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational.

Operation of the drive is at 600 rpm and the load controls the quadrature current amplitude (once steady state operation is achieved), as mentioned earlier. FAST®operation starts at 0.22 s and the torque is initially limited by the 2.5 A quadrature current setting for the speed controller.

The vector plot (see figure 4.275), which corresponds to time 0.5 s, is important, as it shows the scaled (by a factor 500) rotor flux vector ('green'), current vector ('red') and terminal voltage vector ('blue').
The key observation from this vector plot is that the rotor flux vector and current vector (which is due the quadrature current) are orthogonal as required and expected.

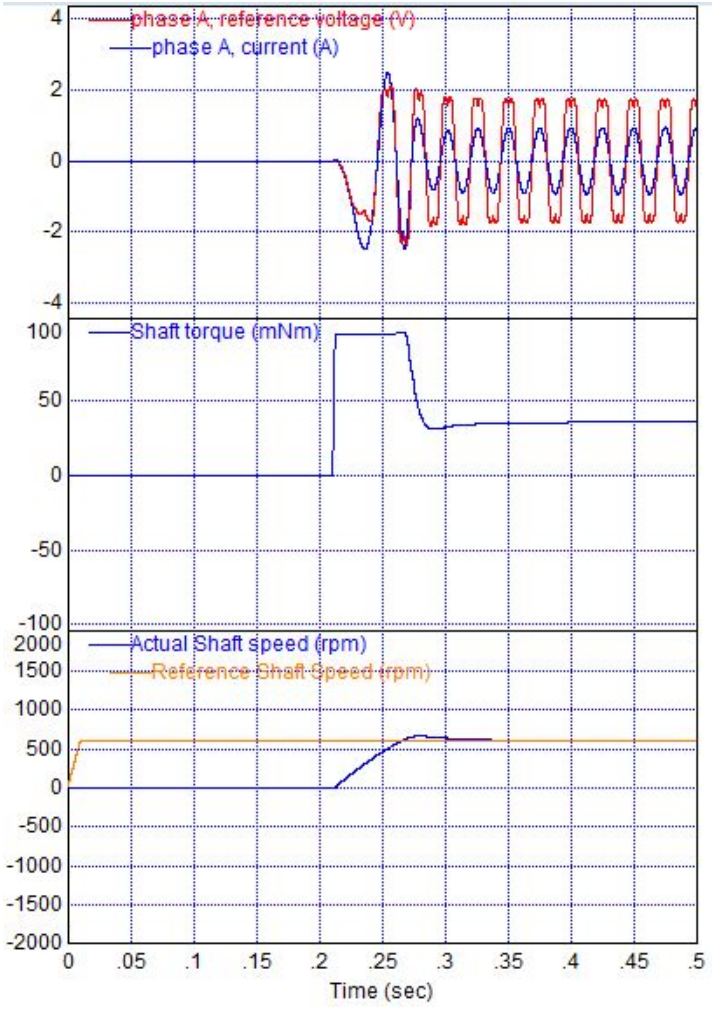


Fig. 4.274 PIL Result: FOC sensorless operation using FAST®with speed control

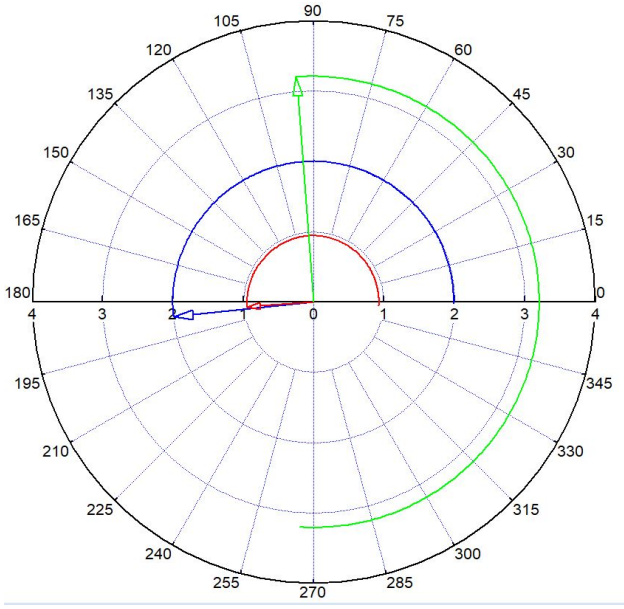


Fig. 4.275 PIL result, PMsyn operation, sensorless FOC-speed control active:vector plot

4.6.3.2 FAST FOC-F069-GANINV:COSIM

Reference files:

- `COSIM-PIL-FOC_PM_FAST_F069-GANINV.vsm`
- `FOC_FAST-PIL_mod.vsm`
- `GAN_INVv12.psimsch`
- `my motor model_v3.psimsch`

The Processor In the Loop (PIL) approach discussed in the previous subsection 4.6.3.1, is designed to evaluate the controller within the target MCU, in this case the F28069, which contains the DLL used to communicate with FAST. In this example the PIL approach is used together with a PSIM co-simulation (COSIM) module, where use is made of a switching converter that is in accordance with the `GANINV` unit, as discussed earlier (see figure 4.10). This implies that effects of using a switching converter on, for example, the shaft torque (resulting in a torque ripple) can be examined.

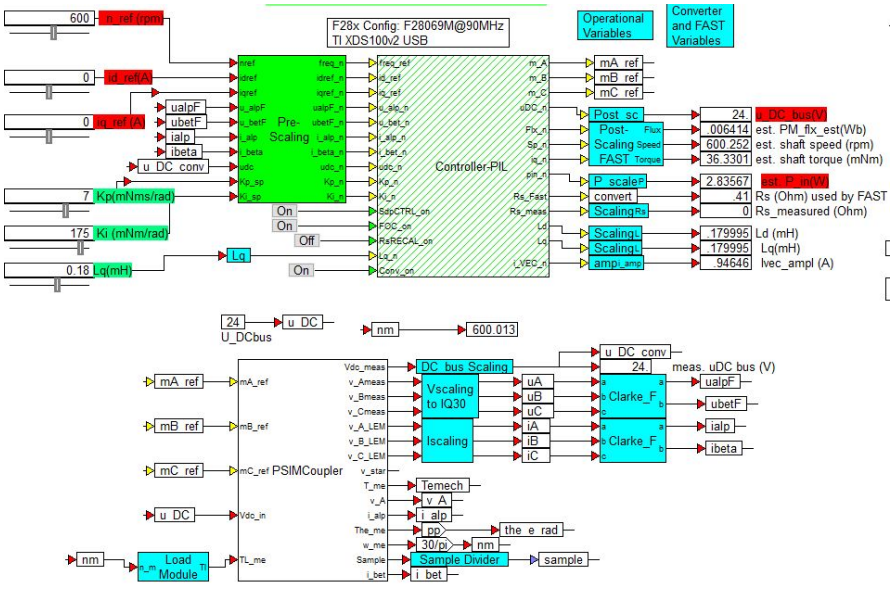


Fig. 4.276 COSIM-PIL model: FOC sensorless operation using FAST®with speed control

The COSIM-PIL model for this example (see figure 4.276), is a screen shot of operation at time 0.5 s with speed control and FAST®active. The corresponding time plot over the period 0→0.5 s (see figure 4.277), shows the voltage waveform across phase A of the motor (scaled by a factor 1/5), phase current,

actual shaft torque (from the IRTF PM model) and reference/measured shaft speed.
Observation of these results, show that FAST® operation starts at 0.22 s and confirms the presence of a torque ripple component due to converter switching. The results are similar to those shown earlier (see figure 4.274), where an 'average voltage per sample converter' was used and same drive conditions.

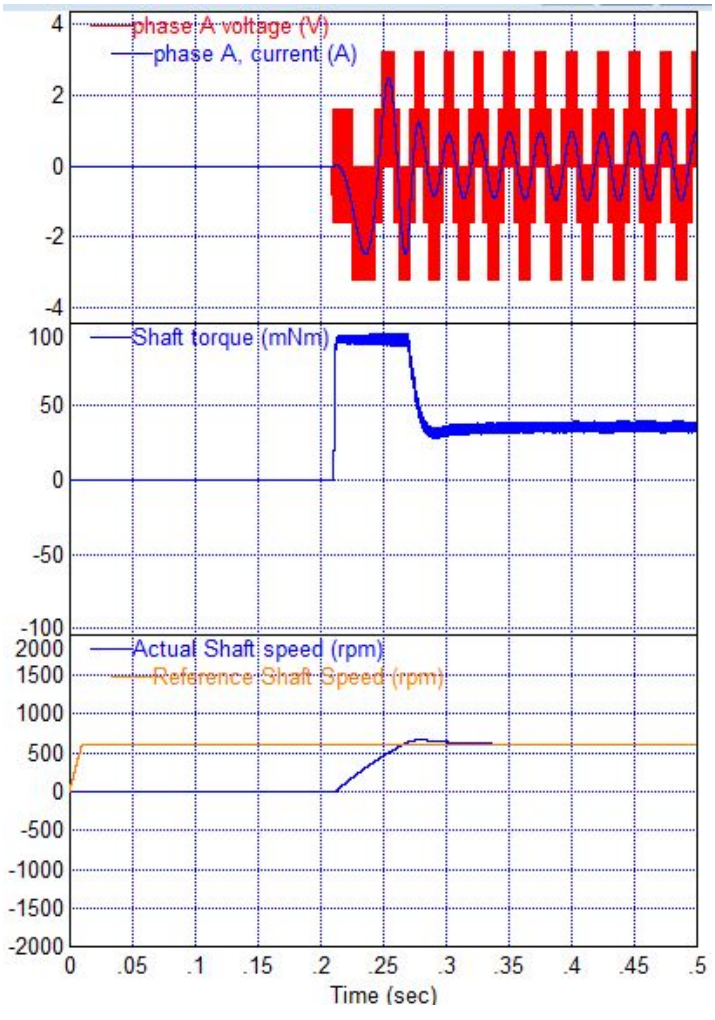


Fig. 4.277 COSIM-PIL Result: FOC sensorless operation using FAST® with speed control

A zoom in of figure 4.277, given in figure 4.278 over a time period of 1 ms shows the typical voltage waveform across a phase of the machine (scaled by a factor $^1/_5$).

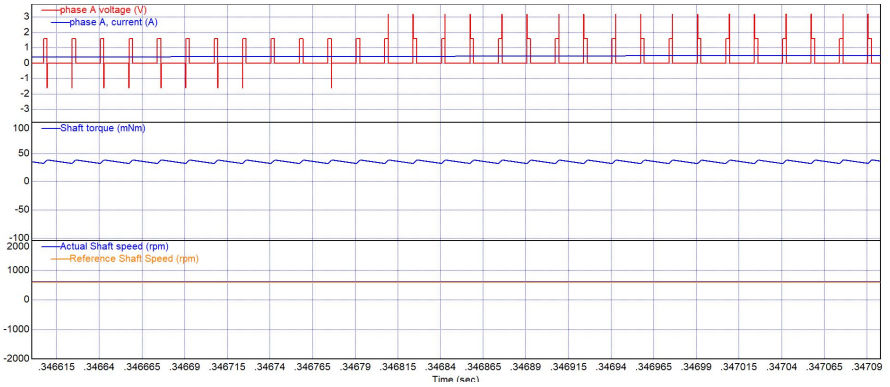


Fig. 4.278 COSIM-PIL Result: FOC sensorless operation using FAST®with speed control, zoom in at time mark 346.6 ms, ΔT =1 ms

Typically the voltage across a phase, will depend on which converter switches are active and this means that the voltage can be zero, ± 8 ,or ± 16 , as is the case here, given a DC bus voltage of 24 V.

A vector plot (see figure 4.279) taken at time 0.26 s (hence 4 ms after FAST®start-up), shows the unity rotor flux axis ('green'), current vector ('red') and measured LPF voltage vector ('blue').
The presence of converter switching is clearly apparent, as may be observed by movement of the current vector endpoint, that has the shape of a triangle formed by two active voltage vectors, the zero vector and the EMF vector [13].

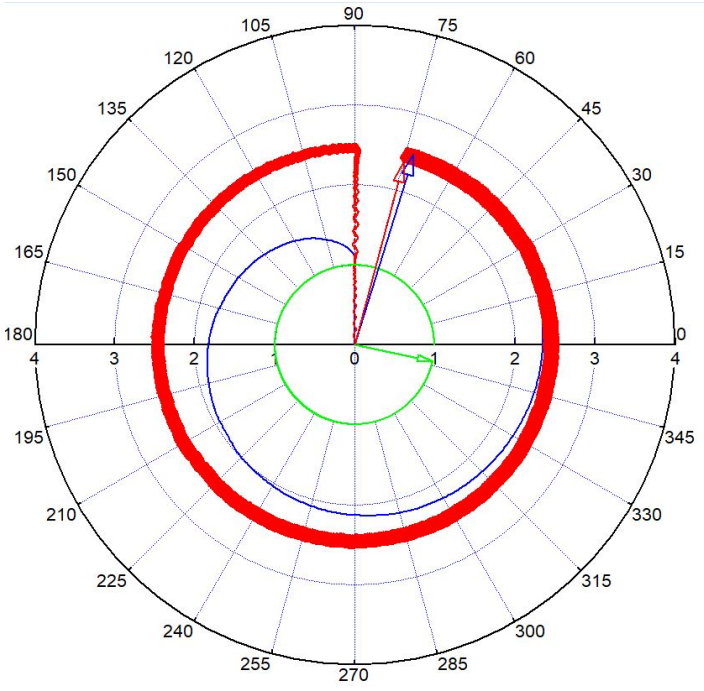


Fig. 4.279 COSIM-PIL Result: FOC sensorless operation using FAST®with speed control, vector plot at time 0.26 s

4.6.3.3 FAST FOC-F069-GANINV:HIL

Reference files:

- `FAST_F069-GANinv-compile.vsm`, for HIL-compile
- `FAST_F069-GANinv-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.290, represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F28069` module), which is connected to a PMsyn Teknic motor[2].

Fig. 4.280 F28069 MCU-GANINV module setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches.
A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration.
The model given in figure 4.281 shows the `FOC_FAST` module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct and quadrature axis reference currents `id`, `iq` and the reference shaft speed `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller. Also added to this example is an input for tuning/setting the inductance used by FAST. It is shown as a 'quadrature inductance', but for the non-salient machine used here the inductance is simply L_s.

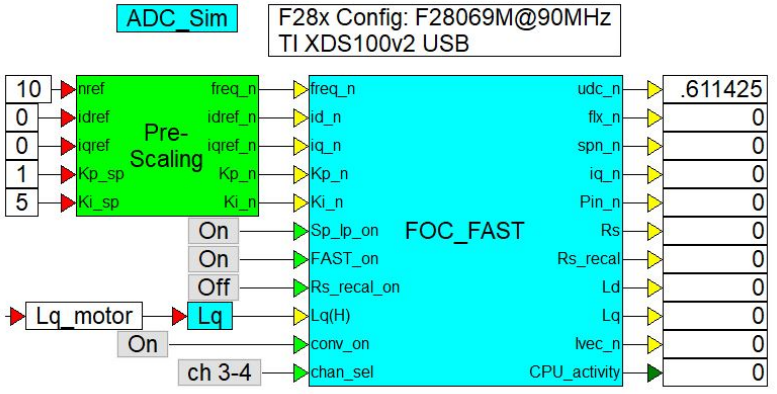


Fig. 4.281 HIL-Compile model of a PMsyn drive with field-oriented sensorless control using FAST®

Five logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed.
These are used as follows:

- `Sp_loop_on`: Speed loop control, when active the shaft speed slider controls the speed, based on the estimated shaft speed generated via FAST®.
- `FAST_on`: Activates sensorless FOC using FAST®. Note when OFF the FAST®algorithm is still operational and this is useful when C&F control is active (both `Sp_loop_on` and `FAST_on` buttons:OFF).
- `Rs_recal_on`: Measures the stator resistance at start-up (when `conv_on` is ON) by injection of a direct axis current (value set in the 'operational variables' dialogue box). The duration of the stator resistance estimation time is set in the `100Hz` module via a time delay setting. It is important to verify that the estimation time is sufficient to reach a steady state resistance estimate.
- `conv_on`: Activates the controller.
- `chan_sel`: Determines which two measurements are shown on the scope unit.

Outputs of the Controller module are:

- `udc_n`: The per unit DC bus voltage, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- `flx_n`: The scaled rotor flux estimate generated by FAST®.
- `sp_n`: A per unit shaft speed estimate `sp_n` generated via FAST®.
- `iq_n`: The measured per unit quadrature current, used to generate a shaft torque estimate by making use of the estimated flux and pole pair number.

- `Pin_n`: A per unit estimate of the motor input power. This variable is also useful as an estimate for the drive input power (bus this ignores the converter losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx {}^{p_{\mathrm{in}}}/_{u_{\mathrm{DC}}}$.
- `Rs`: The per unit stator resistance value used by FAST®.
- `Rs_recal`: The per unit stator resistance value generated by the resistance estimator at start-up (when `Rs_recal_on` is ON). When selected the estimated stator resistance value is transferred to the FAST algorithm on completion of the estimation sequence. When not active the value shown remains zero and is not used by FAST®.
- `Ld`: The direct axis inductance which for non-salient machine will equal the quadrature inductance.
- `Lq`: The quadrature inductance, controlled by the slider input, which is equal to the stator inductance for PMsyn & BLDC machines, as used here.
- `ivec_n`: The measured per unit current vector amplitude.
- `CPU_activity`: Provides the user with a % reading of MCU activity, relative to the controller sampling time.

Also shown in figure 4.281 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module), leads to the set of modules and dialogue boxes (see figure 4.282).

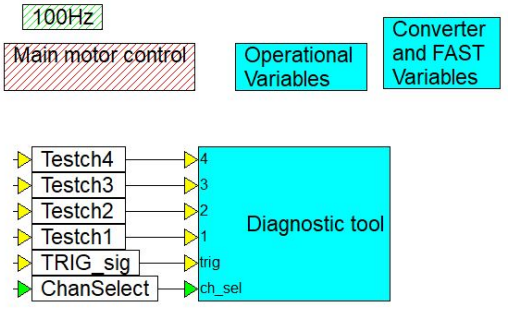


Fig. 4.282 One level down into the FOC-FAST module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.283), is used to assign full scale values for the voltage, current,

frequency and the ramp change for the reference current(s)-frequency(shaft speed). In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the speed controller sampling frequency is typically set lower then used by the main controller. In this case it is set to 1 kHz. These setting are similar to those introduced for FOC with encoder, as discussed in section 4.3.1.4. Added to this dialogue box is the variable `Res_est_current`, which is the DC injection current value used for stator resistance estimation when the option `Rs_recal` is selected.

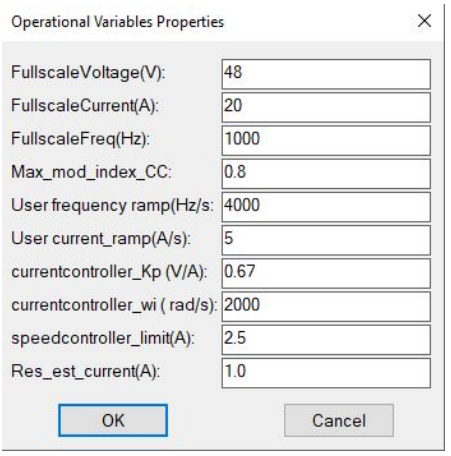


Fig. 4.283 HIL: Dialogue box 'Operational Variables'

The 'Converter and FAST Variables' dialogue box, as given in figure 4.284, defines the scaling used by the Analogue to Digital converter, as discussed earlier (see equation (4.2)).
This module also shows the low-pass filter corner frequency used by the converter for measuring the half-bridge voltages, see figure 4.12) and is an important parameter for FAST®, as it is used to compensate for speed dependant phase angle and amplitude errors of the LPF terminal voltage vector.
Also added to this dialogue module are a number of FAST®parameters namely:

- `F_est/F_sampling`: set the sampling rate used by FAST®relative to the controller sampling rate, typically set to 1.
- `PM_flux)est (Wb)`: an important parameter which represents an estmate for the PM rotor flux in Wb. This value can be estimated, by using the drive in the C&F mode, see the 'Post Analysis' 4.2.1.5). Typically this value should be less or equal to the PM flux value of the machine.
- `Speed filter pole`: corner frequency of the first order LP filter used by the FAST®speed controller.

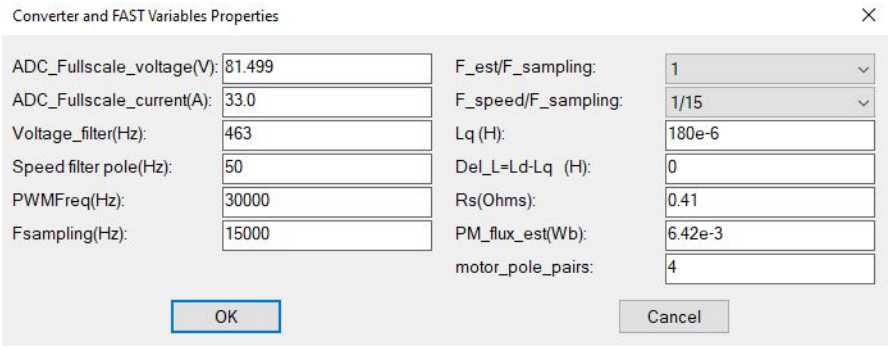


Fig. 4.284 HIL: Dialogue box 'Converter and FAST Variables'

- `Rs (Ohms)` the stator resistance value which is an important FAST®variable at startup of the drive. In this context the EMF threshold speed to shaft ratio has been introduced (see equation (4.23) which for this drive is equal to 3.95 rpm/mNm, hence under no load (typically 10 mNm) the threshold speed (where the EMF equals the voltage drop across the resistance) is equal to 39.5 rpm. However, when operating at rated torque of 100 mNm (set by the speed controller quadrature current) the threshold speed increases to $n_e = 395$ rpm, which shows the importance of accurate stator resistance values.
- `L_q`: the quadrature inductance, which for this non-salient machine is equal to the stator inductance and can be measured using an L-C-R meter (as required to set the current controller gain).
- `Ld-L_q`: The difference between the direct and quadrature inductance, which for PMsyn and BLDC is zero.
- `p`: Pole pair number, which can be found by operating the current in C&F mode and observing the rotational speed of the rotor when, for example, setting the reference shaft speed to 60 rpm (one rotation per second), alternatively for 'outrunner' BLDC machines the number of rotor magnets can be counted and then halved to yield the pole pair number.

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active. Furthermore, this unit generates a number of FAST®settings, which can generally

be left unchanged.

The key component shown in figure 4.189 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit.
Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 4.285. Of these, the 'Drive controller' has been discussed earlier, see figure 4.273 where it was used in phases: PIL and COSIM of the development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 30 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F28069` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present, which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which requires that the `Main motor control` module is executed after every second (for this drive) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

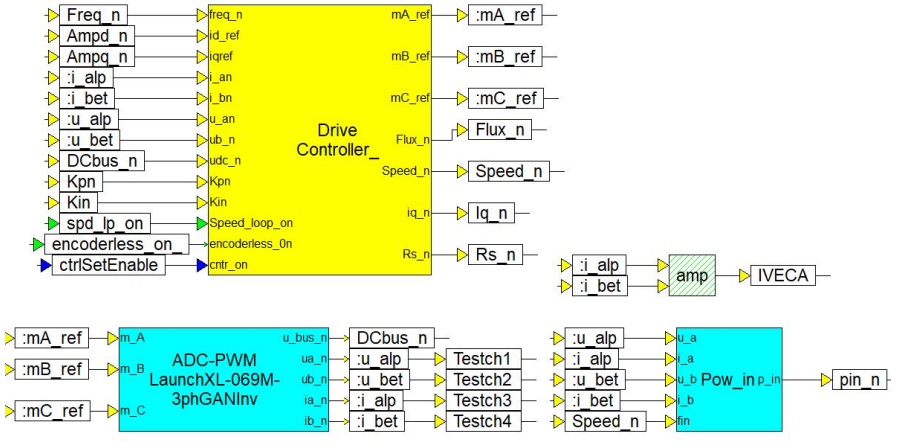


Fig. 4.285 One level down into the 'Main motor controller' module

The `ADC-PWM-LaunchXL-069M-3phGANINV` unit is used generate the PWM signals for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then transformed to

stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β, with outputs `i_alp`, `i_bet` respectively. ADC offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the `100Hz` module) and controlled by said module, as mentioned previously. The per unit DC bus voltage, `DCbus_n_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example (see figure 4.286), and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. Three sliders are used to set the direct/quadrature axis current reference values and the shaft speed. An additional three sliders are present, of which two set the proportional/integral speed control gains, and one is used to set the quadrature inductance value for FAST®.
Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. In addition a set of numerical displays are used to show: estimated PM flux, estimated shaft speed, estimated shaft torque, input power, stator resistance value used by FAST®, measured stator resistance (if this function is activated at start-up), direct and quadrature axis inductance values and current vector amplitude. In addition, a set of gauges are used to show (low-pass filtered): shaft torque, shaft speed, input power and estimated DC supply current.
Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

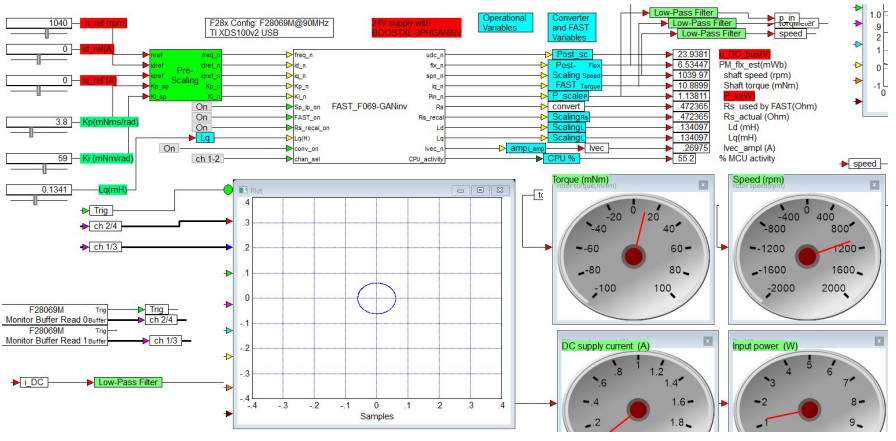


Fig. 4.286 HIL-Debug model, F0049-GANINV of a PMsyn drive with field-oriented control using FAST®

The plot given in figure 4.286, displays the per unit measured LPF vector locus. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus. The shaft speed reference in figure 4.286 is set to 1040 rpm. These results shows operation under encoderless FOC-speed control with the machine operating under no-load conditions.

Post Analysis

In this section an overview of some important tests, that are helpful for commissioning a sensorless drive. Also present in the debug file is a time plot that shows the estimated rotor flux amp normalised to the rotor flux value set in the 'Converter and FAST variables' dialogue box. If the value set in said dialog box is equal to the estimated flux, then the normalised flux shown in figure 4.287 will be 1.

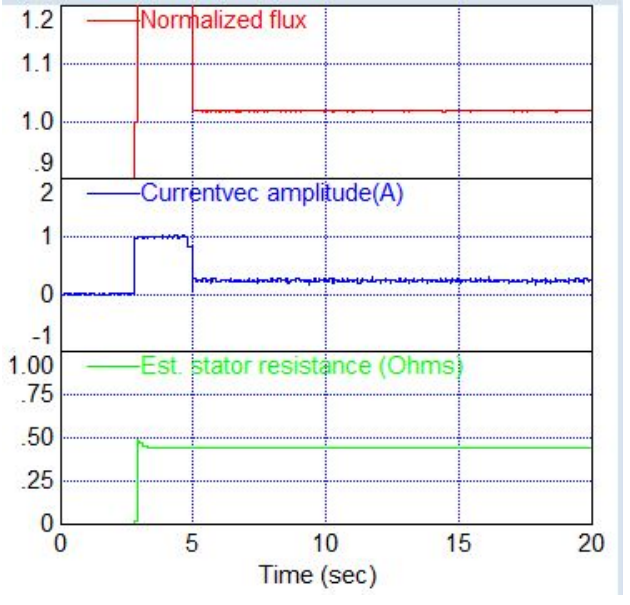


Fig. 4.287 HIL-Debug model, F28069-GANINV of a PMsyn drive: time plot showing Rs estimation at start-up

The time plot given in figure 4.287 also shows the current vector amplitude and the estimated stator resistance (active when the `Rs_recal` function is used).

The example given here, shows stator resistance estimation at start-up when the `Rs_recal` function is active. Initially, the current amplitude moves to 1 A, as expected as this is the value set in the 'operational' dialogue box for Rs estimation. The duration of the Rs estimation is set by the unit `Timer Rs`

located in the `100 Hz` module and should be adjusted to ensure that this routine is switched OFF when a satisfactory steady state resistance value has been reached. Note that after termination of the stator resistance estimation the drive moves to sensorless FOC control as evident by the normalised flux curve. This plot also conveys the message that the rotor flux set in the 'Converter and FAST variables' dialogue box is slightly lower then the value estimated by FAST®.

The same plot is used again to show the impact of direct axis current changes on the normalised flux. Hence the direct axis reference slider is randomly changed (positive and negative) AFTER the start-up sequence (see figure 4.288).

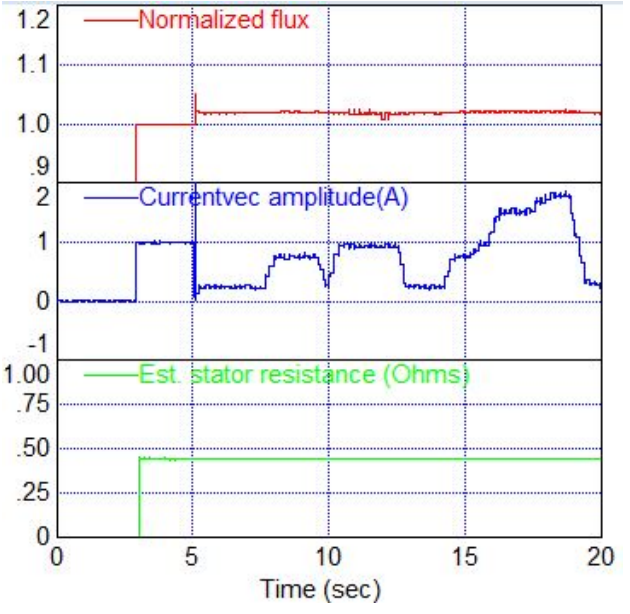


Fig. 4.288 HIL-Debug model, F28069-GANINV of a PMsyn drive: time plot showing effect of direct axis current changes on normalised rotor flux

These results show that direct axis current changes do not have a significant impact. This is expected as the current amplitudes used are relatively low compared to the short circuit current of the machine used, which is $i_{\mathrm{sc}} = 36\mathrm{A}$. Hence the ratio (i_d/i_{sc}) is low (limited to 0.05 in this case) in equation (4.24a), hence the impact of errors between estimated and actual stator inductance is limited.

To check the accuracy of the stator resistance estimate a set of (hand) load changes can be applied which varies the quadrature current, as the machine is operating under FOC speed control. The EMF threshold speed n_e of this

machine for $i_q = 2.5$A (value set for the max speed current reference) is 397 rpm and operation for this example is set to 1000 rpm. The results given in figure 4.289 show the impact of mechanical load changes, which are reflected in current amplitude changes (caused by i_q variations, as $i_d = 0$).

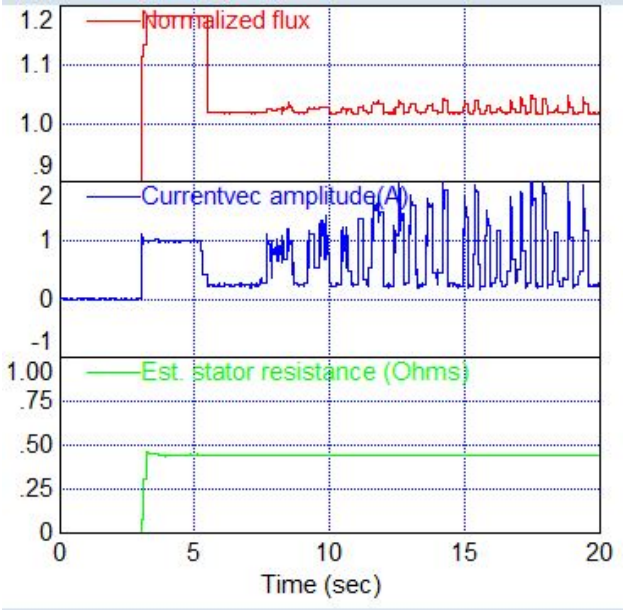


Fig. 4.289 HIL-Debug model, F28069-GANINV of a PMsyn drive: time plot showing effect of quadrature axis current changes on normalised rotor flux

These results show that quadrature axis current changes do not have a significant impact on the flux. This is expected as the voltage drop across the stator resistance is about 40 % when compared to the EMF amplitude, which for the shaft speed used is 2.69 V (can be calculated using $e = {}^{2\pi\, n_m\, p\, \psi_M}/_{60}$). Hence the ratio $({}^{i_q\, Rs}/_{e})$ is approximately 0.4 in equation (4.24b), thus the impact of errors between estimated and actual stator resistance is limited. Furthermore, stator resistance estimation at start-up was also done, which implies that the resistance error will be very small.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

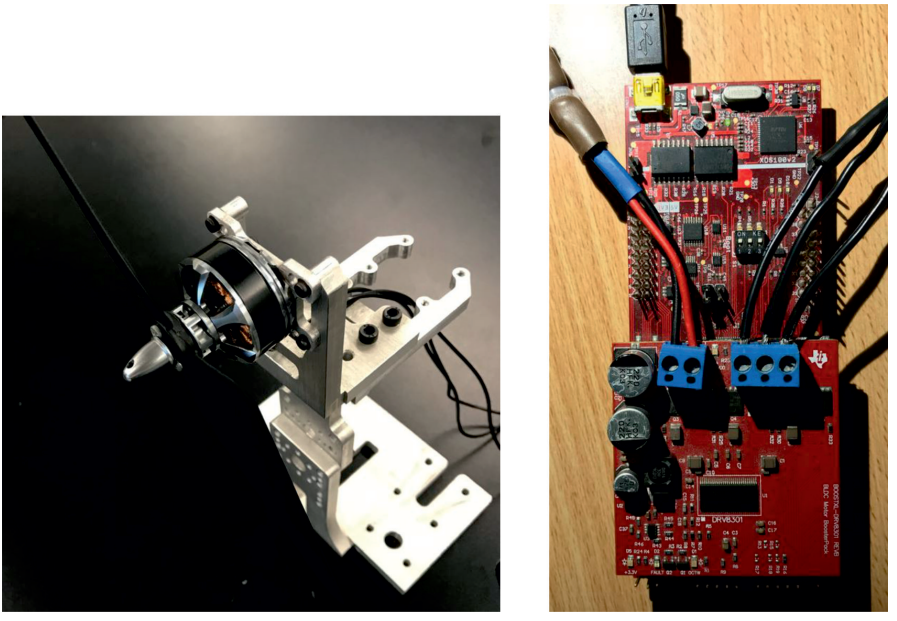
After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.6.3.4 FAST FOC-F069-boostXL8301-drone: HIL

Reference files:

- `D-FAST_F069-boostXL8301-compile.vsm`, for HIL-compile
- `D-FAST_F069-boostXL8301-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) (see figure 4.290), represents the `BOOSTXL-8301` converter (located on the `LAUNCHXL-F28069` module), which is connected to a 'outrunner' BLDC drone 'out-runner' motor[8].
For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.291, shows the `FOC_FAST` module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct and quadrature axis reference currents



(a) Drone setup (b) F28069-boostXL8301

Fig. 4.290 MCU-converter-Drone setup for HIL operation

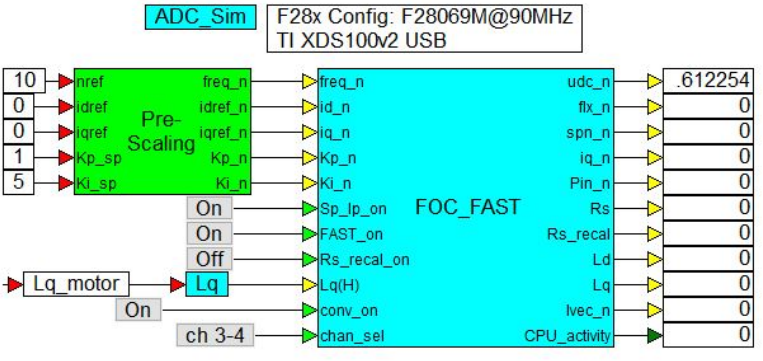


Fig. 4.291 HIL-Compile model of a BLDC drive with field-oriented sensorless control using FAST®

`id`,`iq` and the reference shaft speed `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller. Also added to this example is an input for tuning/setting the inductance used by FAST. It is shown as a 'quadrature inductance', but for the non-salient machine used here the inductance is simply L_s.

Five logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed.
These are defined as follows:

- `Sp_loop_on`: Speed loop control, when active the shaft speed slider controls the speed, based on the estimated shaft speed generated via FASt®.
- `FAST_on`: Activates sensorless FOC using FAST®. Note when OFF the FAST®algorithm is still operational and this is useful when C&F control is active (both `Sp_loop_on` and `FAST_on` buttons:OFF).
- `Rs_recal_on`: Measures the stator resistance at start-up (when `conv_on` is ON) by injection of a direct axis current (value set in the 'operational variables' dialogue box). The duration of the stator resistance estimation time is set in the `100Hz` module via a time delay setting. It is important to verify that the estimation time is sufficient to reach a steady state resistance estimate.
- `conv_on`: Activates the controller.
- `chan_sel`: Determines which two measurements are shown on the scope unit.

Outputs of the Controller module are:

- `udc_n`: The per unit DC bus voltage, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- `flx_n`: The scaled rotor flux estimate generated by FAST®.
- `sp_n`: A per unit shaft speed estimate `sp_n` generated via FAST®.
- `iq_n`: The measured per unit quadrature current, used to generate a shaft torque estimate by making use of the estimated flux and pole pair number.
- `Pin_n`: A per unit estimate of the motor input power. This variable is also useful as an estimate for the drive input power (bus this ignores the converter losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx {}^{p_{\mathrm{in}}}/_{u_{\mathrm{DC}}}$.
- `Rs`: The per unit stator resistance value used by FAST®.
- `Rs_recal`: The per unit stator resistance value generated by the resistance estimator at start-up (when `Rs_recal_on` is ON). When selected the estimated stator resistance value is transferred to the FAST algorithm on completion of the estimation sequence. When not active the value shown remains zero and is not used by FAST®.
- `Ld`: The direct axis inductance which for non-salient machine will equal the quadrature inductance.

- `Lq`: The quadrature inductance, controlled by the slider input, which is equal to the stator inductance for BLDC machine, as used here.
- `ivec_n`: The measured per unit current vector amplitude.
- `CPU_activity`: Provides the user with a % reading of MCU activity, relative to the controller sampling time.

Also shown in figure 4.291 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes shown in figure 4.292.

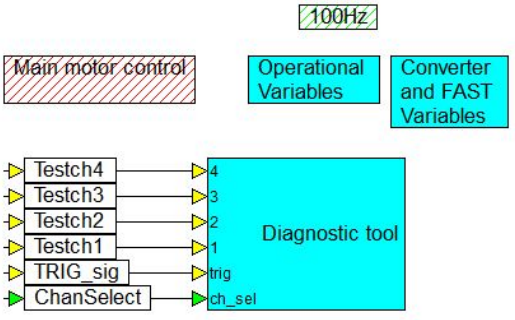


Fig. 4.292 One level down into the FOC-FAST module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.293), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current(s)-frequency (shaft speed). In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the DC current inject level for stator resistance measurement `Rs_est_current(A)` at start-up (when activated) is set in the dialogue box.
The 'Converter and FAST Variables' dialogue box (see figure 4.294), defines the scaling used by the Analogue to Digital converter, as discussed earlier (see equation (4.2)).
This module also shows the low-pass filter corner frequency used by the converter for measuring the half-bridge voltages (see figure 3.13, which shows the `boostXL8301` connected to a brushed DC machine, not a BLDC) and is an important parameter for FAST®, as it is used to compensate for speed dependant phase angle and amplitude errors of the LPF terminal voltage

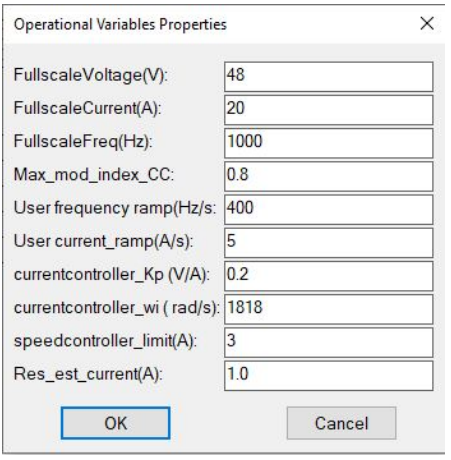


Fig. 4.293 HIL: Dialogue box 'Operational Variables'

vector.

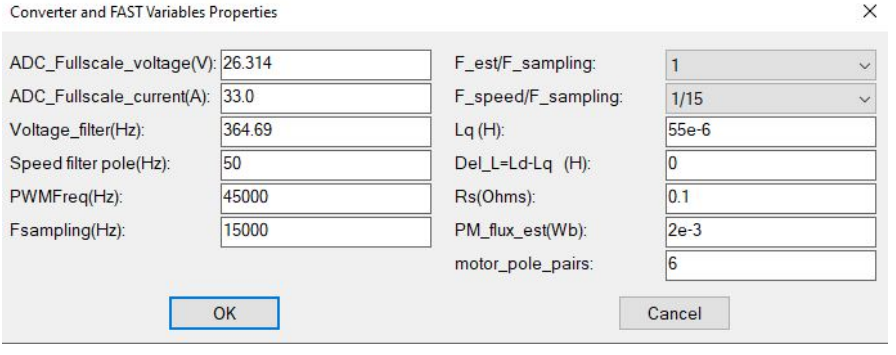


Fig. 4.294 HIL: Dialogue box 'Converter and FAST Variables'

Also added to this dialogue module are a number of FAST®parameters namely:

- `F_est/F_sampling`: set the sampling rate used by FAST®relative to the controller sampling rate, typically set to 1.
- `PM_flux)est (Wb)`: an important parameter, which represents an estimate for the PM rotor flux in Wb. This value can be estimated, by using the drive in the C&F mode (see the 'Post Analysis' 4.2.1.5). Typically this value should be less or equal to the PM flux value of the machine.
- `Speed filter pole`: corner frequency of the first order LP filter used by the FAST®speed controller.

- `Rs (Ohms)` the stator resistance value, which is an important FAST®variable at startup of the drive. In this context the EMF threshold speed to shaft ratio has been introduced (see equation (4.23)), which for this drive is equal to 3.95 rpm/mNm, hence under no load (typically 10 mNm) the threshold speed (where the EMF equals the voltage drop across the resistance) is equal to 39.5 rpm. However, when operating at rated torque of 100 mNm (set by the speed controller quadrature current) the threshold speed increases to $n_e = 395$ rpm, which shows the importance of accurate stator resistance values.
- `L_q`: the quadrature inductance, which for this non-salient machine is equal to the stator inductance and can be measured using an L-C-R meter (as required to set the current controller gain).
- `Ld-L_q`: The difference between the direct and quadrature inductance, which for PMsyn and BLDC is zero.
- `p`: Pole pair number, which can be found by operating the current in C&F mode and observing the rotational speed of the rotor when, for example setting the reference shaft speed to 60 rpm (one rotation per second). In this case an 'out-runner' BLDC drone motor is used, hence the number of rotor magnets can be counted and is equal to 12 in this example, hence the pole pair number is equal to 6.

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active. Furthermore, this unit generates a number of FAST®settings, which can generally be left unchanged.

The key component shown in figure 4.292 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 4.295. Of these, the 'Drive controller' has been discussed earlier (see figure 4.273), where it was used in phases: PIL and COSIM of the development tool-chain (for a different PM machine).
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 45 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: 0.1 μs
- Board in use: `LAUNCHXL-F28069` with `BOOSTXL-8301` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after every third (for this drive) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

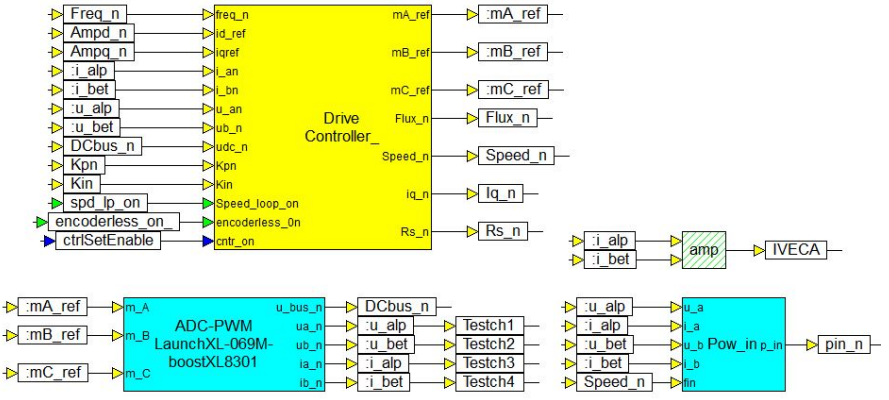


Fig. 4.295 One level down into the 'Main motor controller' module

The `ADC-PWM-LaunchXL-069M-boostXL8301` unit is used generate the PWM signals for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then transformed to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively. ADC offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the `100Hz` module) and controlled by said module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example (see figure 4.296) and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct/quadrature axis current reference values and the shaft speed. Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays.

Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

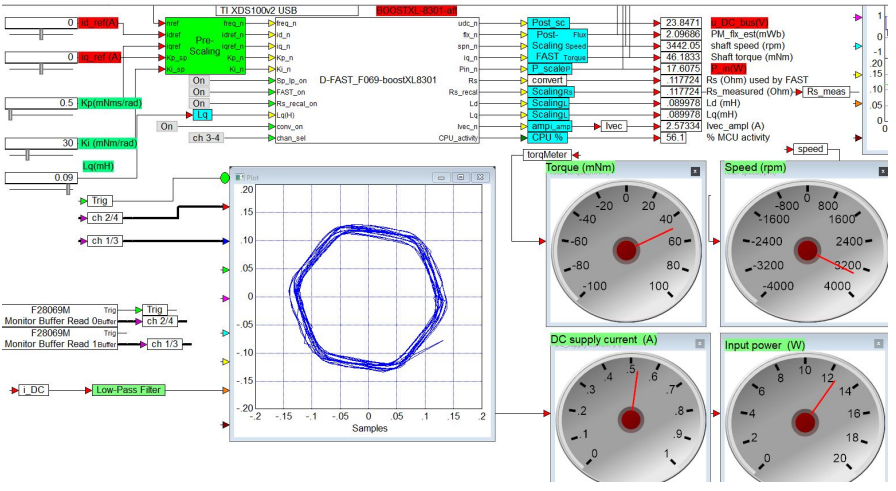


Fig. 4.296 HIL-Debug model, F28069-BOOSTXL8301 of a BLDC Drone drive with field-oriented control using FAST®

The plot given in figure 4.296 displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Note that the current locus is not circular, as is the case for PMsyn machines. A BLDC machine has a sixth harmonic in the EMF, hence the EMF locus has a hexagonal component, which has an impact on the current locus as may be observed from figure 4.296. Selecting `ch 1-2`, will display the per unit LPF voltage vector locus, which is circular. The shaft speed reference (not shown) in figure 4.296 is set to 3440 rpm. These results shows operation under encoderless FOC-speed control with the machine operating under load conditions.

Post Analysis

The EMF threshold speed to shaft torque ratio $\left(\frac{n_e}{T_{me}}\right)$ (calculated using equation (4.23)) of this machine is equal to 3.1 rpm/mNm. Consequently, the threshold shaft speed n_e for the machine with load shown in the example (see figure 4.296) for T_{me} =46 mNm is 142 rpm and operation for this example is set to 3440 rpm. Consequently, the impact of stator resistance errors is not significant. Furthermore, the short-circuit current of this machine is equal to $i_{sc} = 36$A, hence high compared to the quadrature current shown in this example. Therefore, the impact of the voltage across the inductance is relatively low when compared to the EMF amplitude.

It is instructive to estimate the maximum operating speed using the setting and machine parameters for this BLDC machine. The results given in figure 4.297 where generated using file: `Drone-FW_estimator.vsm` and show the estimated maximum shaft speed, shaft torque and output/dissipated power of the drive when operating under maximum quadrature current conditions $i_q^{\max} = 3\text{A}$, with $i_d = 0$. Use of field-weakening by, for example, introducing a

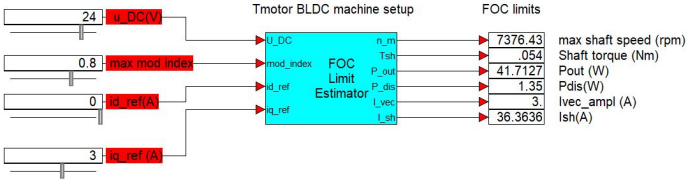


Fig. 4.297 Field-Weakening file used to estimate the maximum shaft speed of the drive

negative direct axis current of $i_d = -3\text{A}$, will have a limited impact given the high quadrature current to short-circuit current ratio of this machine. The estimated increase in the maximum speed of the drive when moving from $i_d = 0$ to $i_d = -3$ A is equal to $7284 \rightarrow 8008$ rpm, as can be found by making use of the file: `Drone-FW_estimator.vsm`.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.6.3.5 FAST FOC-F027-boostXL8301-Turnigy:HIL

Reference files:

- `Turnigy_FAST_F027_boostXL8301-compile.vsm`, for HIL-compile
- `Turnigy_FAST_F027_boostXL8301-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.298 shows the `BOOSTXL-8301` converter (located on the `LAUNCHXL-F28027` module), which is connected to an 'outrunner' BLDC motor[10].

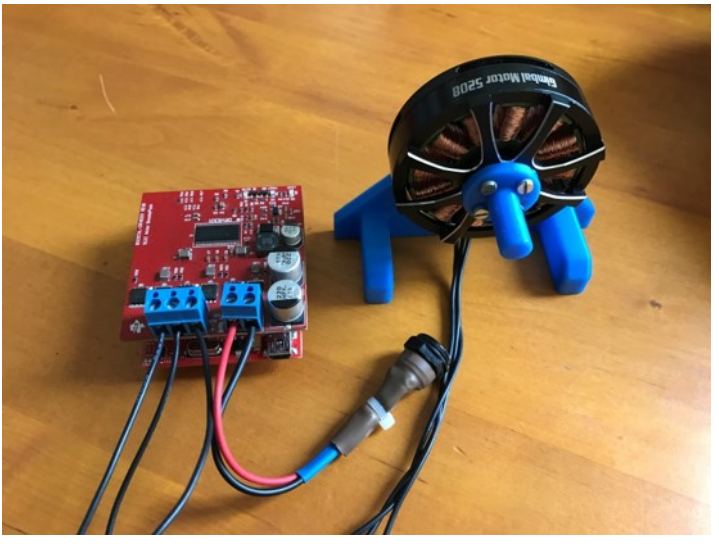


Fig. 4.298 F28027 MCU-boostXL8301 converter-Turnigy setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches.
A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile

stage is to assemble all the modules needed for drive operation with a given controller configuration. Note that for this processor the compiled must be transferred to the target module used in the 'debug' module via (for example) 'UniFlash'[21]. This is required, as for this processor use is made of the Flash memory (not RAM as done for most examples).
The model given in figure 4.299 shows the `FOC_FAST` module, which must be compiled to generate an out-file.

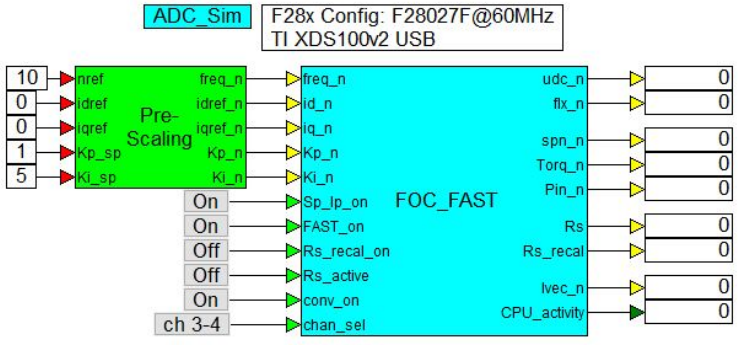


Fig. 4.299 HIL-Compile model of a Turnigy BLDC drive with field-oriented sensorless control using FAST®

Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct/quadrature axis reference currents `id`, `iq` and the reference shaft speed `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller.
Six logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. These are defined, in terms of functionality, as follows:

- `Sp_lp_on`: Speed loop control, when active the shaft speed slider controls the speed, based on the estimated shaft speed generated via FAST®.
- `FAST_on`: Activates sensorless FOC using FAST®. Note when OFF the FAST®algorithm is still operational and this is useful when C&F control is active (both `Sp_loop_on` and `FAST_on` buttons: OFF).
- `Rs_recal_on`: Measures the stator resistance at start-up (when `conv_on` is ON) by injection of a direct axis current (value set in the 'operational variables' dialogue box). The duration of the stator resistance estimation time is set in the `100Hz` module via a time delay setting in module `Timer-Rs`. It is important to verify that the estimation time is sufficient to reach a steady state resistance estimate.
- `Rs_active`: When ON, the estimated resistance generated at start-up will be used by FAST®.

- `conv_on`: Activates the controller.
- `chan_sel`: Determines which two measurements are shown on the plot unit.

Outputs of the Controller module are:

- `udc_n`: The per unit DC bus voltage, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- `flx_n`: The scaled rotor flux estimate generated by FAST®.
- `sp_n`: A per unit shaft speed estimate `sp_n` generated via FAST®.
- `Torq_n`: The shaft torque estimated by FAST®.
- `pin_n`: A per unit estimate of the motor input power. This variable is also useful as an estimate for the drive input power (bus this ignores the converter losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx {}^{p_{\mathrm{in}}}/_{u_{\mathrm{DC}}}$.
- `Rs`: The per unit stator resistance value used by FAST®.
- `Rs_recal`: The per unit stator resistance value generated by the resistance estimator at start-up (when `Rs_recal_on` is ON). When selected (and `Rs_active` is also ON) the estimated stator resistance value is transferred to the FAST algorithm on completion of the estimation sequence. When not active the value shown remains zero and is not used by FAST®.
- `ivec_n`: The measured per unit current vector amplitude.
- `CPU_activity`: Provides the user with a % reading of MCU activity, relative to the controller sampling time.

Also shown in figure 4.299 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. These values will appear in the controller when this file is run. This approach is useful, because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes (see figure 4.300).
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 4.301), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current(s)-frequency. In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the DC current inject level for stator resistance measurement `Rs_est_current(A)` at start-up (when activated) is set in the dialogue box.
The 'Converter and FAST Variables' dialogue box (see figure 4.302), defines the scaling used by the Analogue to Digital converter, as discussed earlier (see equation (4.2)).
This module also shows the low-pass filter corner frequency used by the converter for measuring the half-bridge voltages (see figure 3.13, which shows

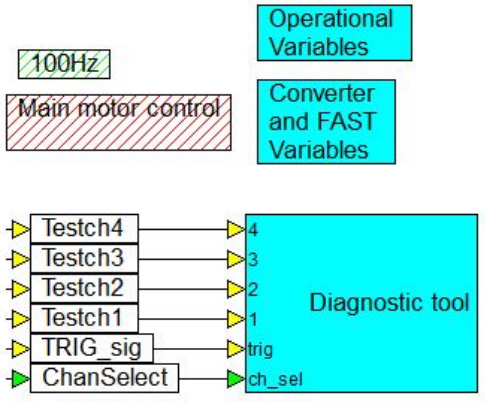


Fig. 4.300 One level down into the FOC-FAST module

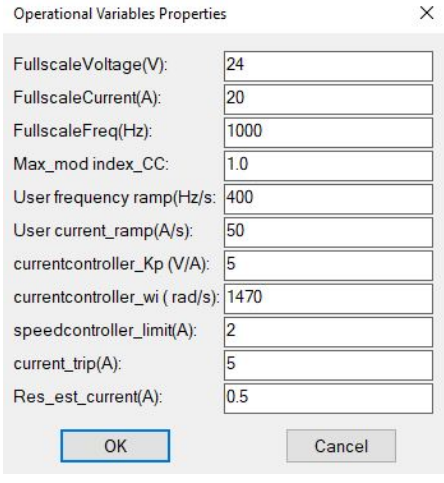


Fig. 4.301 HIL: Dialogue box 'Operational Variables'

the `boostXL8301` connected to a brushed DC machine, not a BLDC) and is an important parameter for FAST®, as it is used to compensate for speed dependant phase angle and amplitude errors of the LPF terminal voltage vector.
Also added to this dialogue module are a number of FAST®parameters namely:

- `F_est/F_sampling`: set the sampling rate used by FAST®relative to the controller sampling rate, typically set to 1.
- `PM_flux)est (Wb)`: an important parameter which represents an estimate for the PM rotor flux in Wb. This value can be estimated, by using the

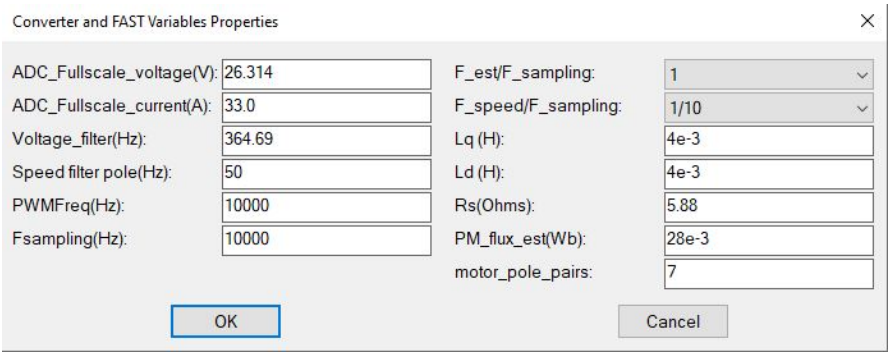


Fig. 4.302 HIL: Dialogue box 'Converter and FAST Variables'

drive in the C&F mode (see the 'Post Analysis' 4.2.1.5). Typically this value should be less or equal to the PM flux value of the machine.

- `Speed filter pole`: corner frequency of the first order LP filter used by the FAST®speed controller.
- `Rs (Ohms)` the stator resistance value which is an important FAST®variable at startup of the drive. In this context the EMF threshold speed to shaft ratio has been introduced (see equation (4.23), which for this drive is equal to 1.08 rpm/mNm, hence under a load (of example 100 mNm) the threshold speed (where the EMF equal the voltage drop across the resistance) is equal to 108 rpm, which shows the importance of accurate stator resistance values for this machine.
- `L_q`: the quadrature axis inductance, which for this non-salient machine is equal to the stator inductance and can be measured using an L-C-R meter (as required to set the current controller gain).
- `Ld`: The direct axis inductance, which for PMsyn and BLDC is equal to `L_q`.
- `p`: Pole pair number, which can be found by operating the current in C&F mode and observing the rotational speed of the rotor when, for example setting the reference shaft speed to 60 rpm (one rotation per second). In this case an 'out-runner' BLDC motor is used, hence the number of rotor magnets can be counted and is equal to 14 in this example, hence the pole pair number is equal to 7.

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC

offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active. Furthermore, this unit generates a number of FAST®settings, which can generally be left unchanged.

The key component shown in figure 4.300 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit.
Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 4.303. Of these, the 'Drive controller' has been discussed earlier, see figure 4.273 where it was used in phases: PIL and COSIM of the development tool-chain (for a different PM machine).
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 10 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0.1 μs
- Board in use: `LAUNCHXL-F28027` with `BOOSTXL-8301` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

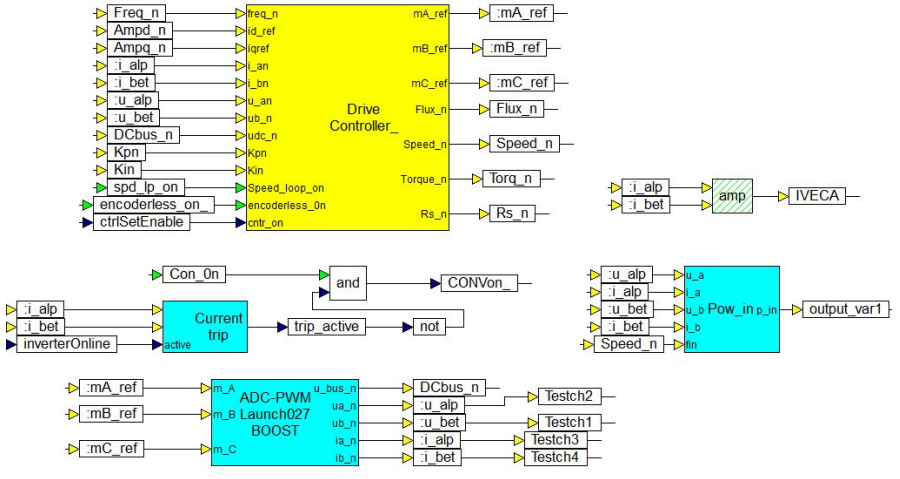


Fig. 4.303 One level down into the 'Main motor controller' module

The `ADC-PWM-LaunchXL-027-BOOST` unit is used generate the PWM signals

for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then transformed to stationary coordinates α, β voltages: `u_alp`, `u_bet`, using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the `100Hz` module) and controlled by said module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example (see figure 4.304) and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above). However, use should be made of, for example, 'UniFlash'[21] to move the compiled `xxx.out` file into Flash memory, as mentioned earlier.
Central to this figure is the Embed 'target execution module' , which executes said out-file. Three sliders are used to set the direct/quadrature axis current reference value and the shaft speed. In addition, two sliders are present for setting the proportional and integral gain of the speed controller.
Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Furthermore, a set of numerical displays have been added to show: estimated PM flux, shaft speed, shaft torque, input power, stator resistance value used by FAST®, measured stator resistance (when activated at startup) and current vector amplitude. Furthermore, two gauges have been added to show (low-pass filtered): shaft speed and shaft torque.
Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

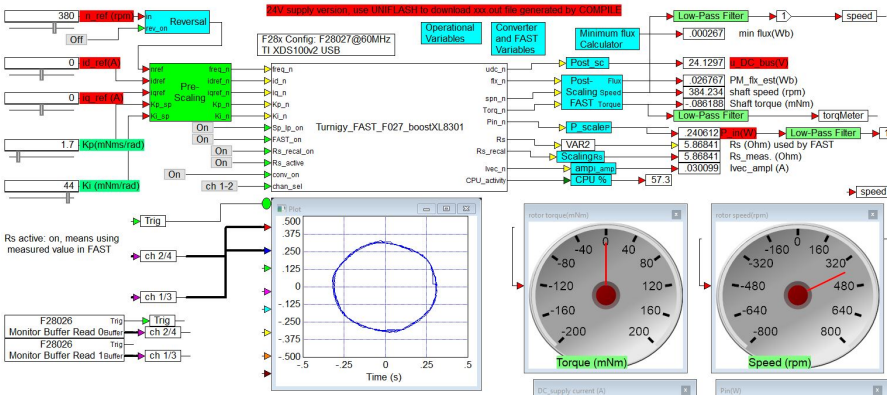


Fig. 4.304 HIL-Debug model, F027-BOOSTXL8301 of a BLDC Turnigy drive with field-oriented control using FAST®

The plot given in figure 4.304 displays the per unit measured LPF voltage vector locus. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus. The shaft speed reference in figure 4.304 is set to 380 rpm. These results shows operation under encoderless FOC-speed control with the machine operating under load conditions.

Post Analysis

It is instructive to estimate the maximum operating speed using the drive settings and machine parameters for this BLDC machine. The results given in figure 4.305, were generated using file: `Turnigy-FW_estimator.vsm` and show estimated maximum shaft speed, shaft torque, output/dissipated power and EMF threshold speed/torque ratio of the drive, when operating with current settings: $i_q = 0.36$A, $i_d = 0$.

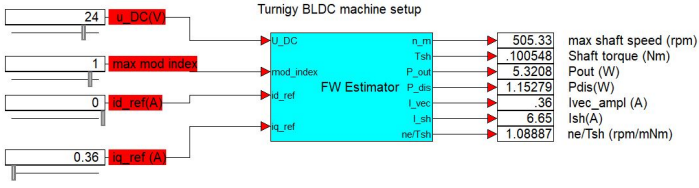


Fig. 4.305 Field-Weakening file used to estimate the maximum shaft speed of the drive

The EMF threshold speed to shaft torque ratio $\left(\frac{n_e}{T_{me}}\right)$ is also given (calculated using equation (4.23)) for this machine and is equal to 1.08 rpm/mNm. Consequently, the threshold shaft speed n_e for the machine with load shown in figure 4.305) for T_{me} =100 mNm is 108 rpm and operation for this example is set to 340 rpm. The short-circuit current of this machine is equal to $i_{sc} = 6.65$A, hence in the same order of magnitude compared to the quadrature current shown in this example. Therefore, the impact of the voltage across the inductance is relatively high when compared to the EMF amplitude.

Use of field-weakening will initially have a positive impact. However, as the negative direct current increases the maximum speed reduces, due to the high voltage drop across the stator resistance, as may be observed by running file `Turnigy-FW_estimator.vsm`, with for example, $i_q = 0$.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

4.7 Sensorless Field Oriented Control PMsyn & BLDC with Field-weakening

The concept of field-weakening for PM machines was introduced earlier for sensored FOC examples (see section 4.5). For sensorless applications the EMF vector shown in the controller q-axis identifies the amplitude and orientation of rotor flux vector, as shown in figure 4.306 (introduced earlier and added here again to facilitate readability).

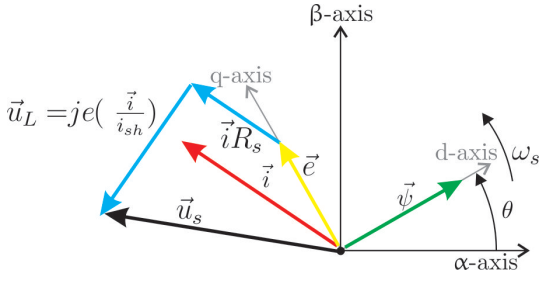


Fig. 4.306 Terminal vector plot: FOC operation PMsyn & BLDC drive with field-weakening

The use of a negative direct axis current reduces the terminal voltage amplitude $\vec{u}_s$, because rotation of the current vector into the second quadrant of the 'dq' coordinate system, also rotates the voltage vectors associated with the voltage across the resistance $\vec{u}_R = \vec{i}\, R_s$ and across the inductance $\vec{u}_L$.
Critical to field-weakening is the amplitude of the vector $\vec{u}_L$, as this is defined by the EMF amplitude times the ratio $\left(|\vec{i}|/i_{\mathrm{sh}}\right)$, as shown in figure 4.306. Consequently, if the short-circuit current i_{sh} of the machine is high relative to the current vector amplitude then the impact of field-weakening is limited, as demonstrated earlier (see the table shown in figure 4.222).

4.7.1 Sensorless Field Oriented Control PMsyn & BLDC with Field-weakening-TI

Purpose of this section is to realize sensorless field-oriented control using a PMsyn or BLDC machine with field-weakening and Texas Instruments hardware. The tool chain, as introduced in figure 2.45, is used as a template for the ensuing subsections, but in this case the tool-chain: PIL→ HIL, will be considered for a specific hardware example. For this purpose, the `F069-GANINV` module connected to a Teknic [2] PMsyn machine has been chosen.

4.7.1.1 FAST FOC-F069-GANINV with Field-weakening: PIL

Reference files:

- `FOC_PM_FAST-FW_F069-GANINV-PIL.vsm`
- `FOC_FAST-FW-PIL_mod.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F28069, which contains the DLL used to communicate with FAST. This implies the use of a target execution module for file: `FOC_FAST-FW-PIL.out` (see figure 4.307).
The target interface module that executes the out-file, generates the modulation indices required for the converter/motor and these components are not affected by use of the PIL. In this example, the IRTF machine model is set to operate in PMsyn mode (not BLDC, given the use of a PM synchronous machine).
Generation of the out file is based on the PIL module (see figure 4.308), which must be compiled using the command: `toolbox>codegen`.
Moving one level into the PIL module `FOC-FAST-FW-PIL` shows a 'Drive Controller' (see figure 4.309).
Added to said controller is a module `LED_ctrl`, which is used to flash an LED

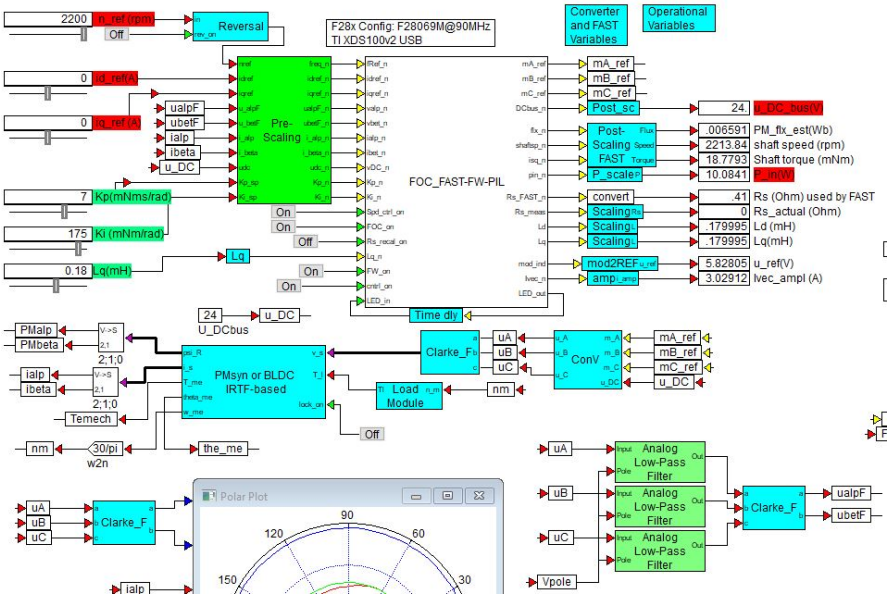


Fig. 4.307 Field-oriented torque/speed control using FAST®with field-weakening: PIL operation

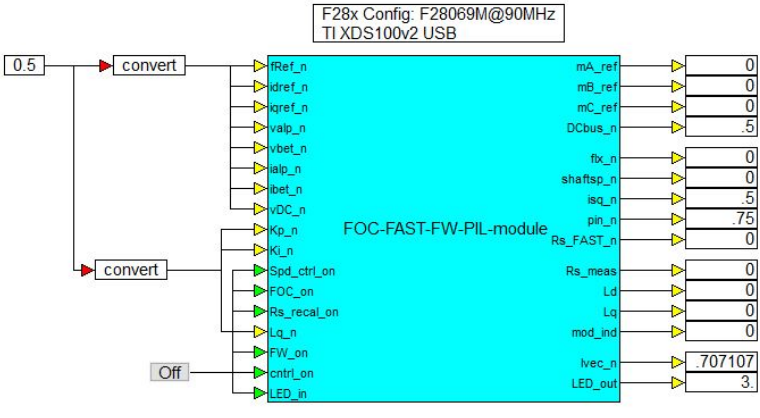


Fig. 4.308 Module:'FOC-FAST-FW-PIL'

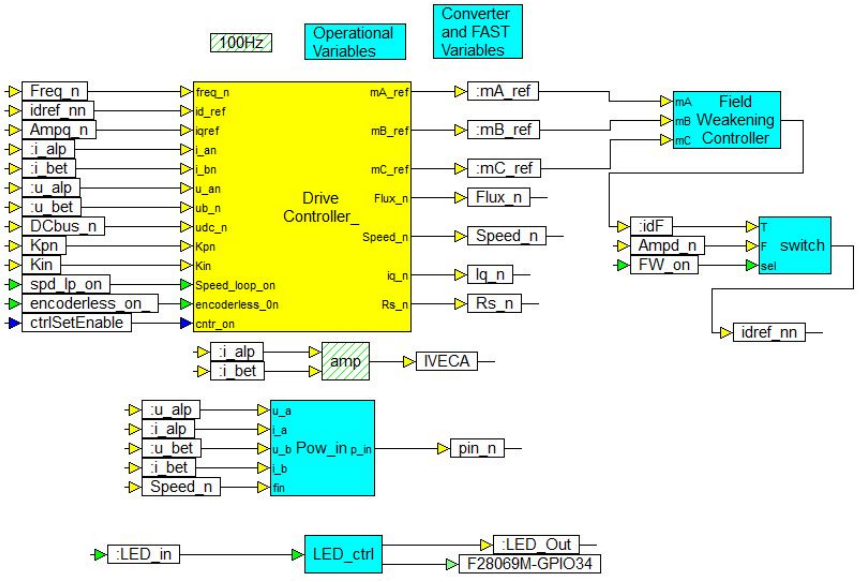
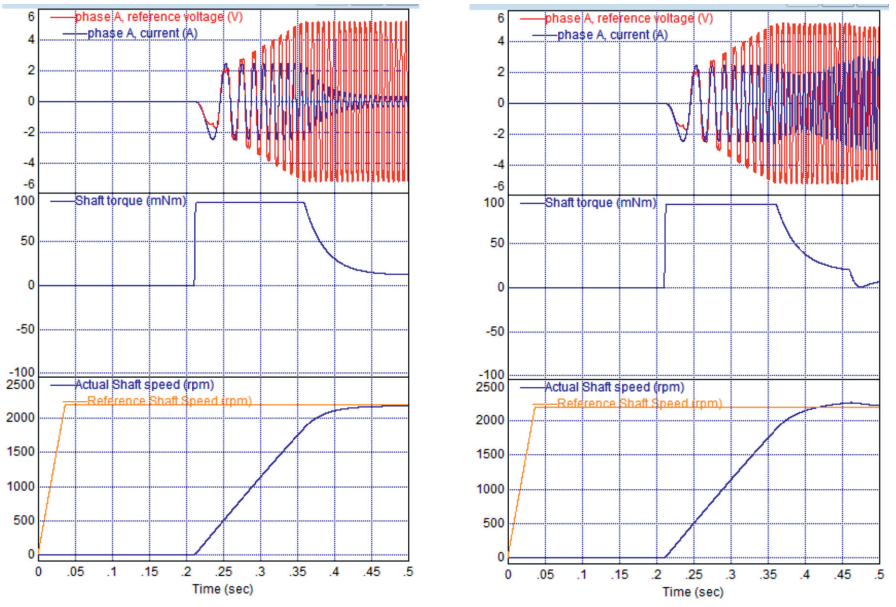


Fig. 4.309 Content PIL module, Field-oriented torque/speed control with FAST® and field-weakening

on the MCU board to show that the PIL control is operational.
A `Field weakening Controller` has been introduced, which makes use of the three phase modulation indices, to generate the required direct axis current reference when field-weakening control is on (variable `FW_on` logic 1). Furthermore, the modules responsible for input power measurement `Pow_in` and current vector amplitude `amp` are located outside the 'Drive Controller'. The drive controller module is as discussed earlier (see figure 4.273).

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational.

The speed reference for the drive is judicially set to 2200 rpm and the quadratic load curve is to set to provide 10 mNm at 2000 rpm. FAST® operation starts at 0.22 s and the torque is initially limited by the 2.5 A quadrature current setting for the speed controller. Plot(a) of figure 4.310, represents operation with field-weakening OFF and clearly shows that the quadrature current controller switches to voltage control once the maximum modulation index of 0.5 (which corresponds to 6 V in this case) is reached. Consequently, the drive shaft speed is unable to reach the reference speed. Plot(b) of figure 4.310, represents operation with field-weakening ON and field-weakening becomes active (as seen by an increase in the phase current) once the maximum modulation index of 0.5 (which corresponds to 6 V in this case) is reached. With field-weakening



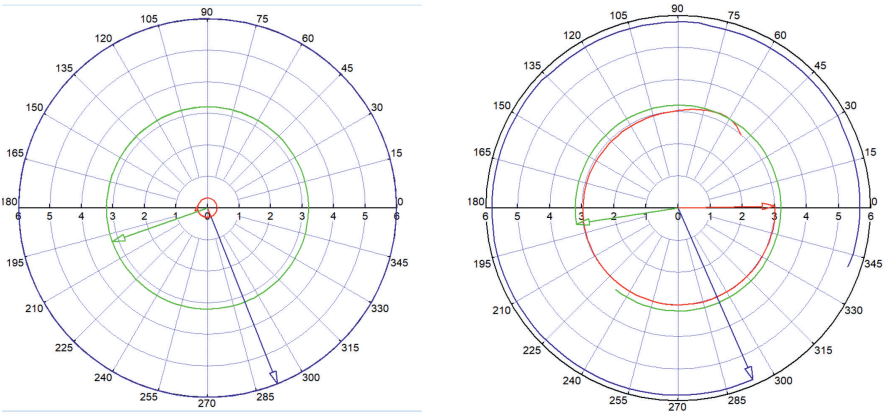
(a) Field-weakening OFF (b) Field-weakening ON

Fig. 4.310 PIL Result: FOC sensorless operation using FAST®with speed control and field-weakening OFF/ON

active the drive speed does reach the reference speed. However, the impact of field-weakening is limited because the ratio of current vector amplitude/PM short-circuit current is low for the machine used in this drive.

The vector plot (see figure 4.311), which corresponds to time 0.5 s, shows the scaled (by a factor 500) rotor flux vector ('green'), current vector ('red') and terminal voltage vector ('blue') for operation with and without field-weakening.

With field weakening OFF (figure 4.311(a)) the current vector is orthogonal to the flux vector and dictated by the applied load which in this case is virtually no-load. The terminal voltage vector amplitude is 6 V as expected, because the quadrature current controller of the drive is operating under voltage control. Note that the phase voltage amplitude under these conditions is a factor $\sqrt{3}/2$ lower, given the use of space vector modulation (as discussed in section 2.1.5). With field weakening ON (figure 4.311(b)) the current vector has a large -3 A direct axis current component and a small quadrature current component, dictated by the applied load, which in this case is virtually no-load. The terminal vector voltage amplitude is slightly less then 6 V as expected. Operation in this case is on the limit of current control (for the quadrature controller).



(a) Field-weakening OFF (b) Field-weakening ON

Fig. 4.311 PIL result, PMsyn operation, sensorless FOC-speed control active: vector plot, with and without field-weakening

4.7.1.2 FAST FOC-F069-GANINV with Field-weakening: HIL

Reference files:

- `FAST_FW_Gal_F-069-GANINV-compile.vsm`, for HIL-compile
- `FAST_FW_Gal_F-069-GANINV-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 4.312 represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F28069` module) which is connected to a PMsyn Teknic motor[2].

Fig. 4.312 F28069 MCU-GANINV setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches.
A compile stage, is used to generate file: `XXX.out`, that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 4.313 shows the `FOC_FAST-FW` module, which must be compiled to generate an out-file.

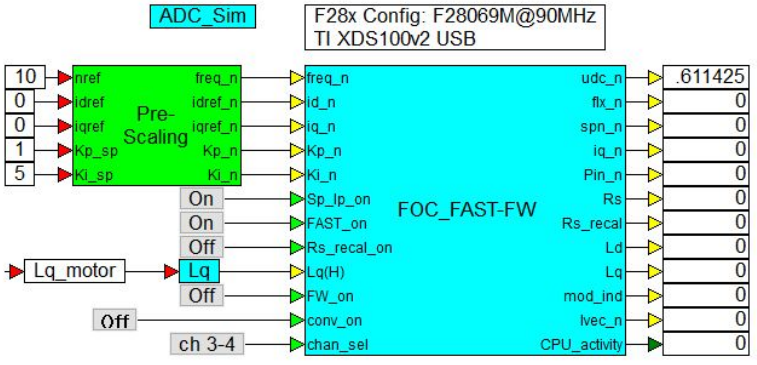


Fig. 4.313 HIL-Compile model of a PMsyn drive with field-oriented sensorless control using FAST®

Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct and quadrature axis reference currents `id`, `iq` and the reference shaft speed `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller. Also added to this example is an input for tuning/setting the inductance used by FAST. It is shown as a 'quadrature inductance', but for the non-salient machine used here the inductance is simply L_s .
Six logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed.
These are used as follows:

- `Sp_loop_on`: Speed loop control, when active the shaft speed slider controls the speed, based on the estimated shaft speed generated via FAST®.
- `FAST_on`: Activates sensorless FOC using FAST®. Note when OFF the FAST®algorithm is still operational and this is useful when C&F control is active (both `Sp_loop_on` and `FAST_on` buttons:OFF).

- `Rs_recal_on`: Measures the stator resistance at start-up (when `conv_on` is ON) by injection of a direct axis current (value set in the 'operational variables' dialogue box). The duration of the stator resistance estimation time is set in the `100Hz` module via a time delay setting. It is important to verify that the estimation time is sufficient to reach a steady state resistance estimate.
- `FW_on`: Activates field-weakening, in which case the direct axis reference slider is non-functional.
- `conv_on`: Activates the controller.
- `chan_sel`: Determines which two measurements are shown on the scope unit.

Outputs of the Controller module are:

- `udc_n`: The per unit DC bus voltage, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- `flx_n`: The scaled rotor flux estimate generated by FAST®.
- `sp_n`: A per unit shaft speed estimate `sp_n` generated via FAST®.
- `iq_n`: The measured per unit quadrature current, used to generate a shaft torque estimate by making use of the estimated flux and pole pair number.
- `pin_n`: A per unit estimate of the motor input power. This variable is also useful as an estimate for the drive input power (bus this ignores the converter losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx {}^{p_{\mathrm{in}}}/_{u_{\mathrm{DC}}}$.
- `Rs`: The per unit stator resistance value used by FAST®.
- `Rs_recal`: The per unit stator resistance value generated by the resistance estimator at start-up (when `Rs_recal_on` is ON). When selected the estimated stator resistance value is transferred to the FAST algorithm on completion of the estimation sequence. When not active the value shown remains zero and is not used by FAST®.
- `Ld`: The direct axis inductance which for non-salient machine will equal the quadrature inductance.
- `Lq`: The quadrature inductance, controlled by the slider input, which is equal to the stator inductance for PMsyn & BLDC machines, as used here.
- `mod_ind`: The modulation index vector, used to calculate the amplitude of the terminal voltage vector.
- `ivec_n`: The measured per unit current vector amplitude.
- `CPU_activity`: Provides the user with a % reading of MCU activity, relative to the controller sampling time.

Also shown in figure 4.313 is an 'ADC-Sim' module, which is used to provide artificial inputs (as pre-set in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after

selecting module) leads to the set of modules and dialogue boxes (see figure 4.314).

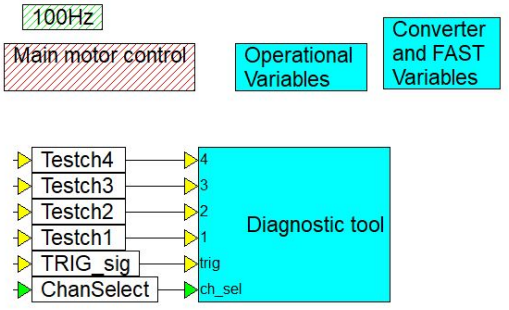


Fig. 4.314 One level down into the FOC-FAST module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'operational Variables' dialogue box (see figure 4.315), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current(s)-frequency. In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the speed controller sampling frequency is typically set lower then used by the main controller. In this case it is set to 1 kHz. These setting are similar to those introduced for FOC with encoder, as discussed in section 4.3.1.4. Added to this dialogue box is the variable `Res_est_current`, which is the DC injection current value used for stator resistance estimation when the option `Rs_recal` is selected. Furthermore, a variable `FW Max_Id` is introduced, which set the maximum negative direct axis current used by the field-weakening controller.
The 'Converter and FAST Variables' dialogue box, as given earlier (see figure 4.284), defines the scaling used by the Analogue to Digital converter, as discussed earlier (see equation (4.2)) and the primary FAST®settings.
This module also shows the low-pass filter corner frequency used by the converter for measuring the half-bridge voltages and is an important parameter for FAST®, as it is used to compensate for speed dependant phase angle and amplitude errors of the LPF terminal voltage vector.

A 'Diagnostic tool' module is used to buffer two variables, which can be subsequently displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

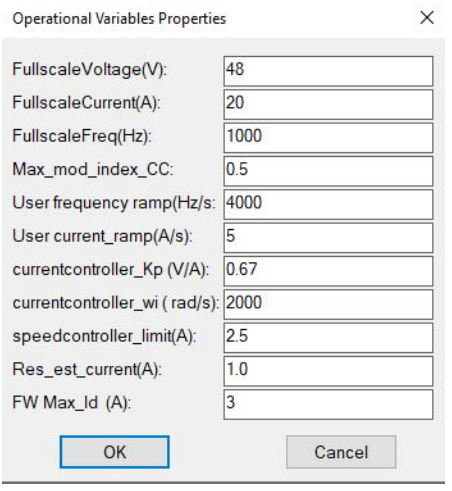


Fig. 4.315 HIL: Dialogue box 'Operational Variables'

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (State machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active. Furthermore, this unit generates a number of secondary FAST®settings, which can generally be left unchanged.

The key component shown in figure 4.314 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules (see figure 4.316). Of these, the 'Drive controller' has been discussed earlier (see figure 4.273), where it was used in phases: PIL and COSIM of the development tool-chain. The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 30 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F28069` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present, which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after every second (for this drive) EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

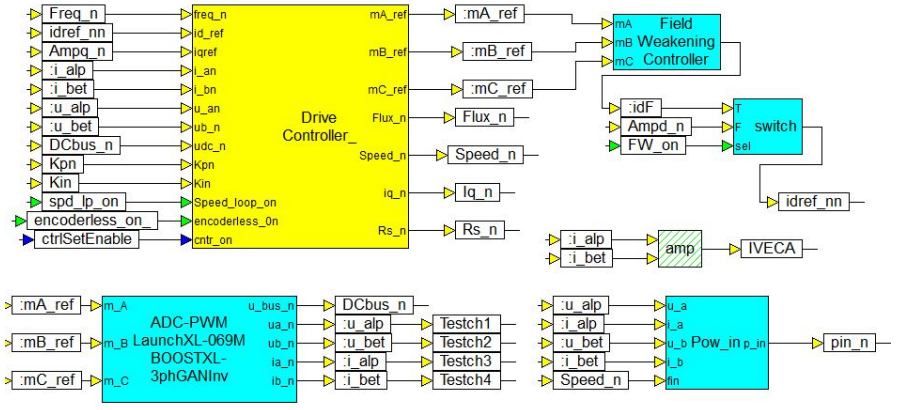


Fig. 4.316 One level down into the 'Main motor controller' module

The `ADC-PWM-LaunchXL-069M-3phGANINV` unit is used generate the PWM signals for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then transformed to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively. ADC offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the `100Hz` module) and controlled by said module, as mentioned previously. The per unit DC bus voltage `DCbus_n`, completes the ADC measurement sequence. The `Field-weakening Controller` added to this module takes over control of the direct axis reference current, when field-weakening is selected (logic variable `FW_on`= 1).

The HIL-debug/run tool-chain component is the operational component of this example (see figure 4.317) and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct/quadrature axis current reference values and the shaft speed. Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. In addition, a set of numerical display modules are present which show: estimated shaft speed, shaft torque, PM flux, direct axis inductance, quadrature axis inductance (as used by FAST®), stator resistance used by FAST®, measured resistance at start-up (if activated), terminal voltage vector amplitude and current vector ampltude. Furthermore, four gauges are present, which show (low-pass filtered): shaft speed, shaft torque, input power and DC supply current.

Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

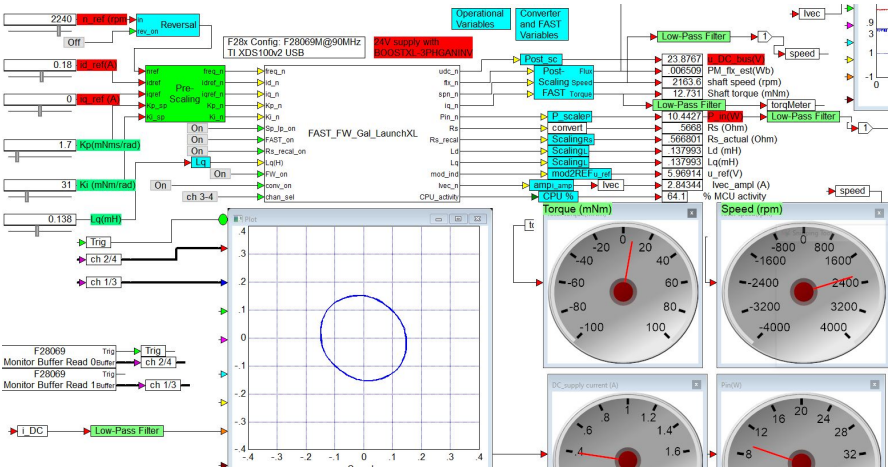


Fig. 4.317 HIL-Debug model, F28069-GANINV of a PMsyn drive with field-oriented control using FAST® with field-weakening option active

The plot (see figure 4.317), displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit LPF voltage vector locus. The shaft speed reference in figure 4.317 is set to 2240 rpm. These results shows operation under encoderless FOC-speed control with the machine operating under no-load conditions and field-weakening active.
Observation of the results show that the quadrature current controller is operating under voltage control and the maximum speed achievable (with field-weakening active) is 2163 rpm (with a maximum modulation setting of 0.5), which corresponds to a terminal voltage of 6 V, as is shown on the numerical display. With field-weakening OFF, the maximum speed (with the same speed reference setting used with field-weakening active) reached is 2100 rpm, hence field-weakening has little impact because the short-circuit current of the machine is much higher then the stator current amplitude. This fact is also reflected in the theoretical table shown earlier for this machine (see figure 4.222).

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

Chapter 5
Hybrid Stepper Drives

This chapter is concerned with the control of hybrid stepper drives, so the developmental tool-chain: 'Simulation' (SIM) and 'Hardware in the Loop'(HIL), is discussed for Texas Instruments (TI) based hardware examples. The drive control options for Hybrid drives considered are: 'Voltage & Frequency Control', and sensorless (no encoder) 'Field Oriented Control' (FOC). An example of a sensorless FOC, using FAST[23] will be discussed.

The overall drive set-up, as given in figure 5.1, shows the two-phase motor connected to a three-phase converter. Important in this case is to assign a 'common' terminal, in this case '`C`', which is connected to both phases of the motor, as may be observed from figure 5.1.

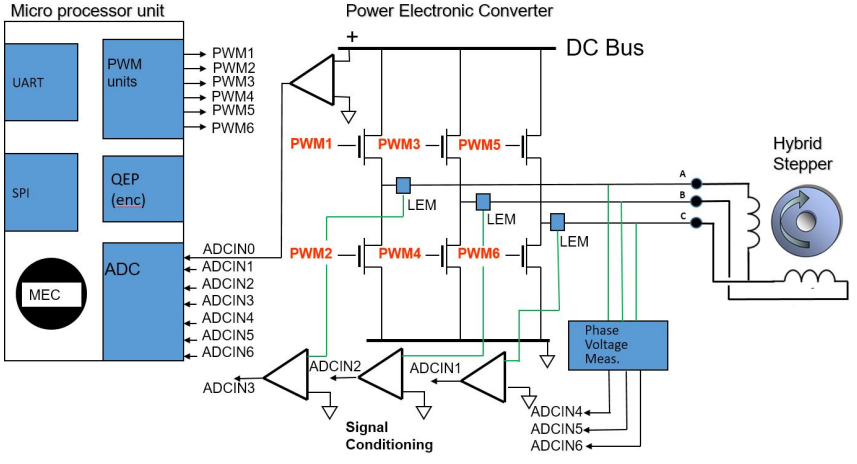


Fig. 5.1 Two-phase hybrid stepper motor connected to a three-phase converter

Current measurement for FOC is done with the aid of the LEM sensors, linked

to terminals 'A' and 'B' (the LEM assigned to terminal 'C' is not used here). Voltage measurement of the three converter outputs is done with the 'Phase Voltage Meas.' unit and is required for sensorless FOC. A micro processor unit is used to control the power electronic converter. Use of software Embed is made to generate the 'MEC' (Micro Embedded Code), which in turn is based on the control algorithms to be discussed in ensuing sections.

The use of a three-phase converter with a two phase machine simplifies drive operation (when compared to using to 'H'-bridges' and most importantly increases the maximum modulation index of the controller to $\sqrt{2}$. This implies that the operating speed range of the drive is increased (assuming of course that the motor can handle the higher phase voltages). This approach, shown here for the hybrid stepper, has also been applied to 'single-phase' (which is named after the supply source used) induction machines [14].

5.1 Voltage & Frequency Control

In this case the objective is to create a controller that can generate a rotating voltage space vector (see figure 5.2) for a 2-phase hybrid stepper. For simplicity the reference vector, $\vec{u}^{\text{ref}}$, is tied to the 'd-axis' of a rotating 'd,q' coordinate system (hence u_q is set to zero for convenience), of which the user can set the rotational speed defined by the reference frequency f^{ref}. The amplitude of the reference vector $\vec{u}^{\text{ref}}$ can also be set by the user, where care should be taken, as the currents encountered will be voltage and frequency dependent.

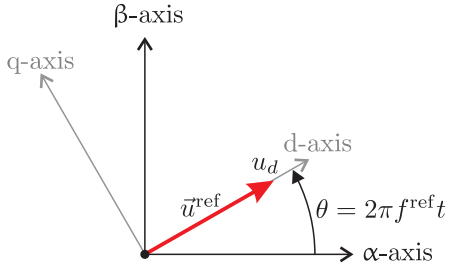


Fig. 5.2 Vector diagram: open loop voltage & frequency control

The angle between the 'd-axis' and stationary 'α-axis', is defined as θ, which is the integral of said reference frequency (in rad/s). Note that the angle θ does NOT represent the orientation of the flux vector, as these examples are concerned with open loop voltage & frequency control. Consequently, the PM flux vector will be aligned with the current vector, when operating under

no-load (as is the case here). Note also that these examples are based on the use of either Texas Instruments or ST Micro hardware.

5.1.1 Voltage & Frequency Control-TI

The purpose of this section is to realize voltage & Frequency control of a hybrid stepper machine using Texas Instruments hardware. At the same time this type of example is used to make a functional check on the current and voltage measurements, and overall functioning of the drive board and software. As such this section provides a set of 'foundation' examples, which will substantially reduce the development time required. The reason being that users can simply redesign or modify the controller to their needs, as the hardware setup phase for HIL's provided is in place.

5.1.1.1 Voltage & Frequency control-Stepper-F069-GANINV: SIM

Reference file: `V&F_069_GANINV-STEP-SIM.vsm`

This example considers the simulation of a two-phase PM Hybrid motor [11] connected to a Texas Instruments F28069M GANINV converter, as shown in figure 5.3. The aim is to examine operation of the drive using a fixed point controller, that will subsequently be used in the HIL implementation.

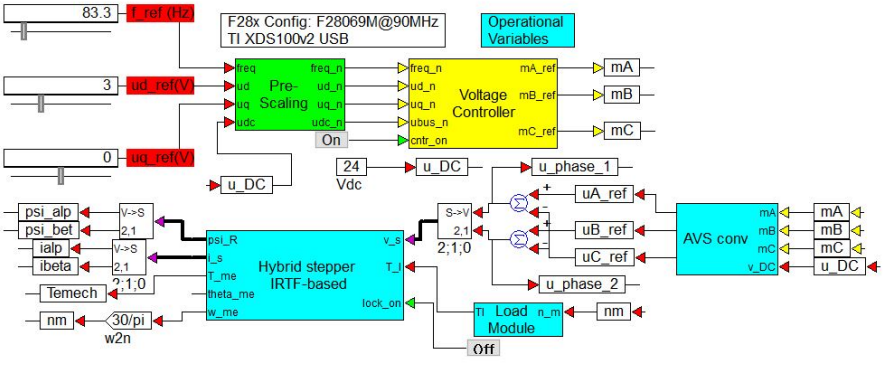


Fig. 5.3 Voltage & Frequency control: simulation

Drilling into the 'voltage controller' reveals the controller structure shown in

figure 5.4. Outputs of the controller are the three modulation indices that control the three converter half bridges.

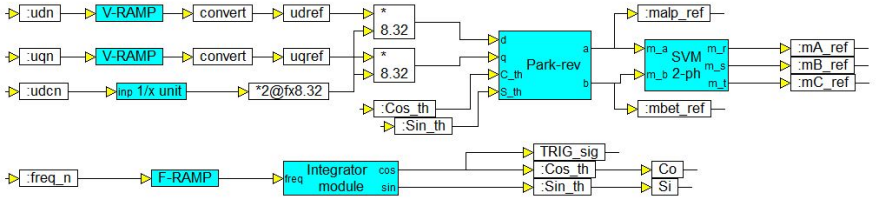


Fig. 5.4 Voltage controller structure

For all controllers considered, the aim is to generate the modulation indices that are required by the PWM units. Consequently, it is important to convert the reference voltage to a modulation index using expression (5.1).

$$m^{\mathrm{ref}} = \frac{2\,u^{\mathrm{ref}}}{u_{\mathrm{DC}}} \tag{5.1}$$

This requires access to the measured DC bus voltage u_{DC}, shown as `udcn`(see figure 5.4). The space vector modulation 'SVM 2-ph' unit is specifically designed for 2-phase operation, see figure 5.5 and is based on a concept introduced earlier for single phase induction machines [14].

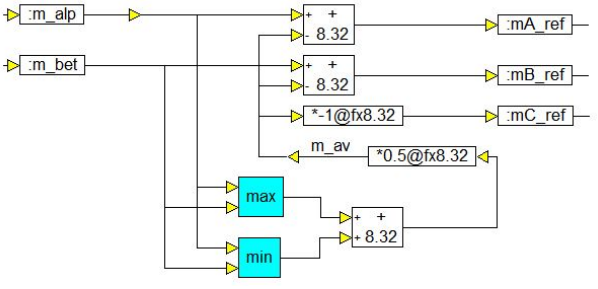


Fig. 5.5 Two-phase space vector modulation unit

Inputs to the SVM module are the modulation indices, `m_alp` and `m_bet` (which are the modulation indices in stationary α, β coordinates). The two-phase SVM modulator generates the average `m_av` of said two signals, and

this is subtracted from the input signals to generate the two phase modulation indices `mA_ref` and `mB_ref`. The third phase modulation reference `mC_ref` is simply the inverse of the average signal `m_av`.
An example showing the relationship between input and output modulation indices (see figure 5.6), underlines operation of the two-phase SVM module.

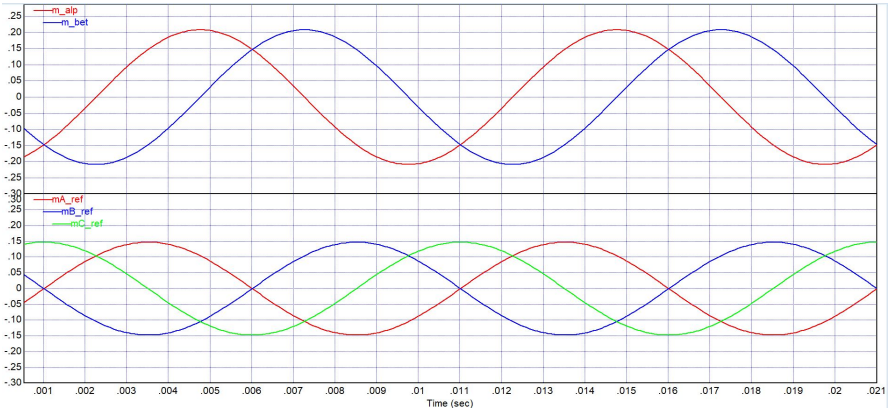


Fig. 5.6 Two-phase space vector modulation: input and output waveforms

An important observation is that the modulation index range for the three-output modulation indices is 40 % lower then the two-phase input modulation indices. Consequently, the maximum modulation indices that can be used by the controller is 1.4 (for regular three-phase converter operation this value is 1.15). This will allow FOC drive operation at higher speeds, as will be discussed shortly.

The voltages generated by the converter, `u_A_ref`,`u_B_ref`,`u_C_ref` , are found by multiplying the modulation indices `mA`, `mB`, `mC` by half the DC bus voltage as executed in module `AVS conv` (see figure 5.3). These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables. The aim of the switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sub chapter 2.1.5). The two phases of the hybrid motor are connected between terminals `A-C` and `B-C` respectively, and consequently each phase sees the difference between these two signals. This is shown in the simulation by the 'summation modules' that provide the two α, β voltages to the motor.
An IRTF based model of the stepper is used, which is almost identical to that of an IPM machine, the difference is that the IRTF torque calculation is now based on the use of a two phase machine (hence the factor $3/2$ is removed from the IRTF torque equation). The dialog box for the stepper [11] used here, as given in figure 5.7 shows the PM salient nature of this machine. Note that

a larger inertia (then specified by the manufacturer) was used to model the presence of a load.

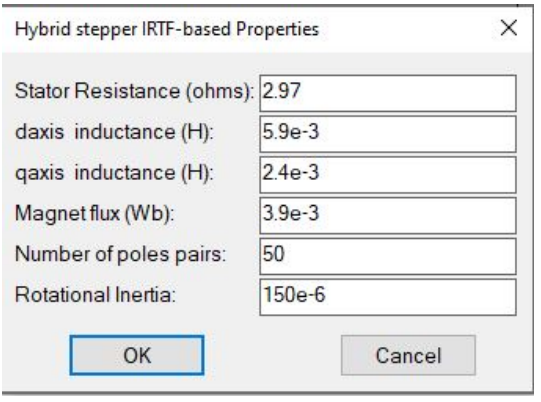


Fig. 5.7 IRTF Hybrid Stepper dialog box

A quadratic load is introduced, which provides a value of 1.0 mNm at a shaft speed of 1000 rpm.

The results shown in figure 5.8, confirm the presence of voltage and frequency control. Note that at start up, ramp functions are used, which determine the rate of reference voltage and frequency variable changes, as set in the 'Operational Variables' dialogue box (to be discussed in the following subsection). The frequency reference was judicially set to 83.3 Hz, which corresponds to a shaft speed of 100 rpm, given that the stepper has a pole pair number equal to 50 .
Of importance is to consider the vector plot (see figure 5.9), which shows the orientation of the unit d-axis ('green'), voltage vector ('blue'), PM flux vector ('brown' and scaled by a factor 500) and current vector ('red') at time 0.25 s. Observation of the voltage vector confirms that it is indeed aligned with the d-axis and its amplitude is 3.0 V, as expected because `uq_ref` is set to zero. Furthermore, the current vector lags the voltage vector, which is expected due to the inductive nature of the machine. Note also that the PM flux vector is aligned with the current vector, under no-load conditions.

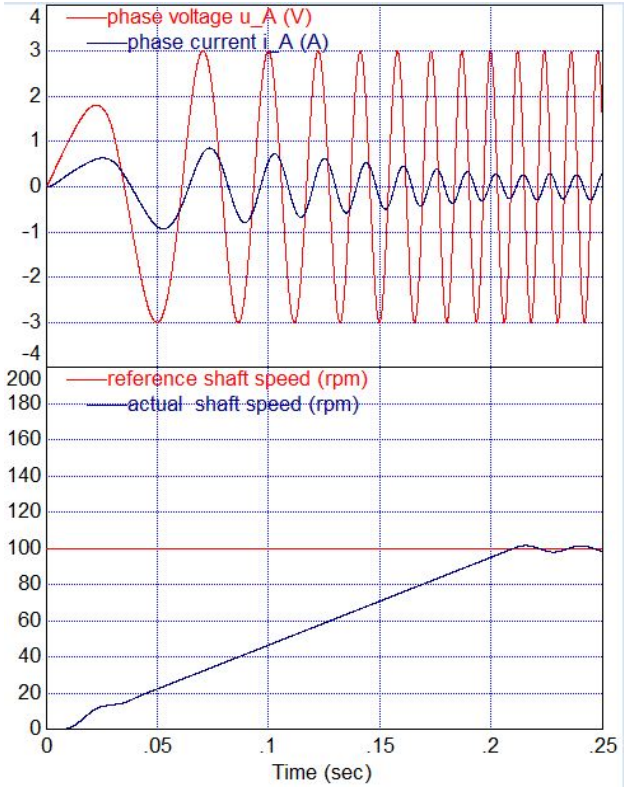


Fig. 5.8 SIM result, hybrid stepper operation, voltage & frequency control

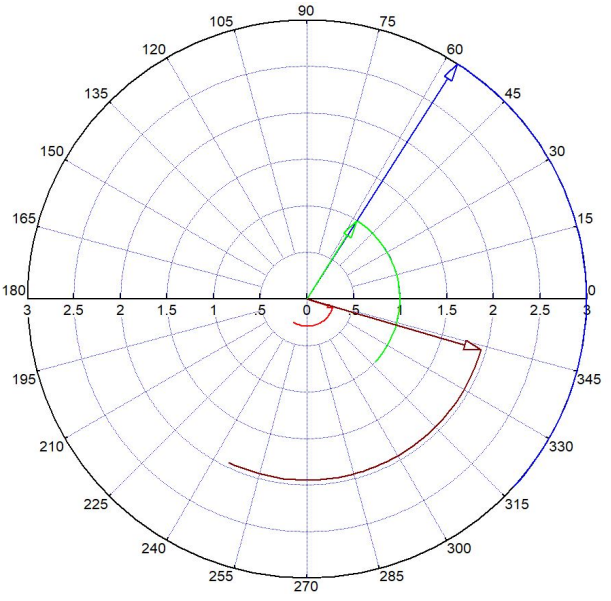


Fig. 5.9 SIM result, hybrid stepper operation operation, voltage & frequency control: vector plot

5.1.1.2 Voltage & Frequency control-Stepper-F069-GANINV: HIL

Reference files:

- `V_F28069-ST-GANINV-compile.vsm`, for HIL-compile
- `V_F28069-ST-GANINV-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), shown earlier (see figure 4.26), represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F28069M` module), which is connected to a 2-phase hybrid stepper motor [11].
For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 5.10 shows the 'Open loop voltage controller' module, which must be compiled to generate an out-file .

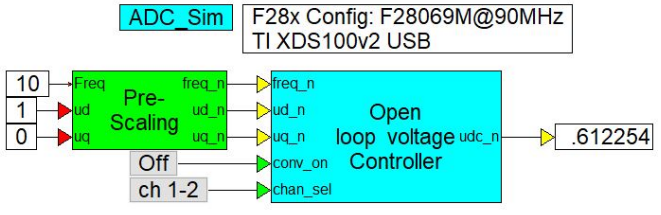


Fig. 5.10 HIL-Compile model of a stepper drive with voltage & frequency control

Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct and quadrature reference voltages `ud`, `uq` and the frequency `Freq`, which are converted to 'per unit' values in the 'pre-scaling' module.
Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed.
Output of the Controller module is the per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
Also shown in figure 5.10 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Hence these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module

are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to a set of modules and dialogue boxes shown in figure 5.11.

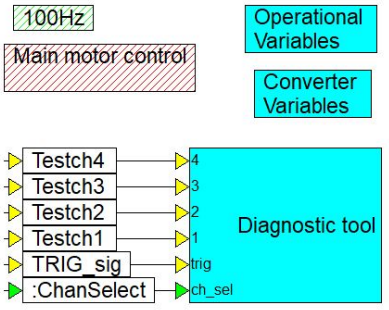


Fig. 5.11 One level down into the 'Open loop Voltage Controller' module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 5.12), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage-frequency.

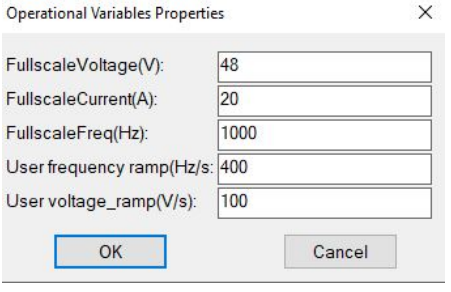


Fig. 5.12 HIL: Dialogue box 'Operational Variables'

The 'Converter Variables' dialog box (see figure 5.13), defines the scaling used by the Analogue to Digital converter, as discussed earlier (see equation (4.4)).

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed

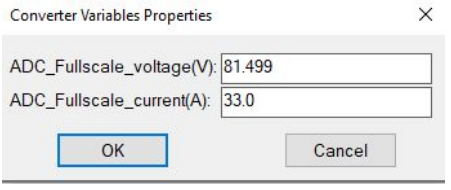


Fig. 5.13 HIL: Dialogue box 'Converter Variables'

above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component (see figure 5.11) is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, as shown in figure 5.14. Of these, the 'Voltage controller' has been discussed previously (see figure 5.4), where it was used in phase: SIM, of the development tool-chain. The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module.
This module is configured with the following basic settings :

- PWM frequency: 15 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F28069M` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.
Moving one level down the `ADC-PWM` unit (see figure 5.15), shows the `PWM unit BOOSTXL-GANINV` that controls the converter switches based on the modulation indices generated by the controller (see figure 5.14).
The `ADC BOOSTXL 3PH-GANINV` unit is used to measure the three low-pass filtered half-bridge voltages and these are then transformed to stationary coordinates α, β voltages: `u_alp_n`, `u_bet_n` using the two `IQ30` 'sum' units, as required to calculate the per unit voltages across the two phases of the stepper motor. The two measured phase currents `ialp_n`, `ibet_n` are those of the stepper and are already in stationary α, β coordinates.

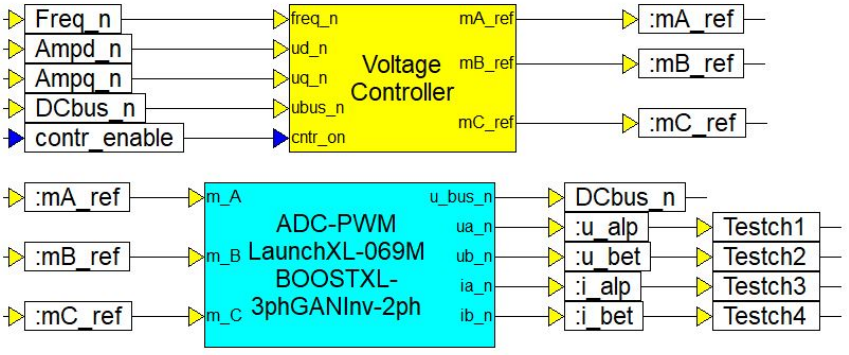


Fig. 5.14 One level down into the 'Main motor controller' module

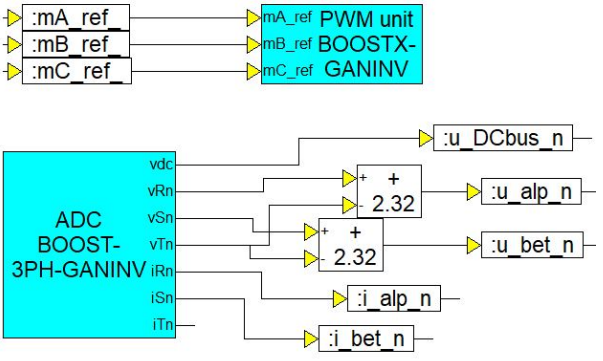


Fig. 5.15 One level down into the 2 phase 'ADC-PWM' module

ADC offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `u_DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example (see figure 5.16), and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct/quadrature voltage reference values and the electrical frequency. Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input)

diagnostic signals. The plot given in figure 5.16, displays the per unit LPF

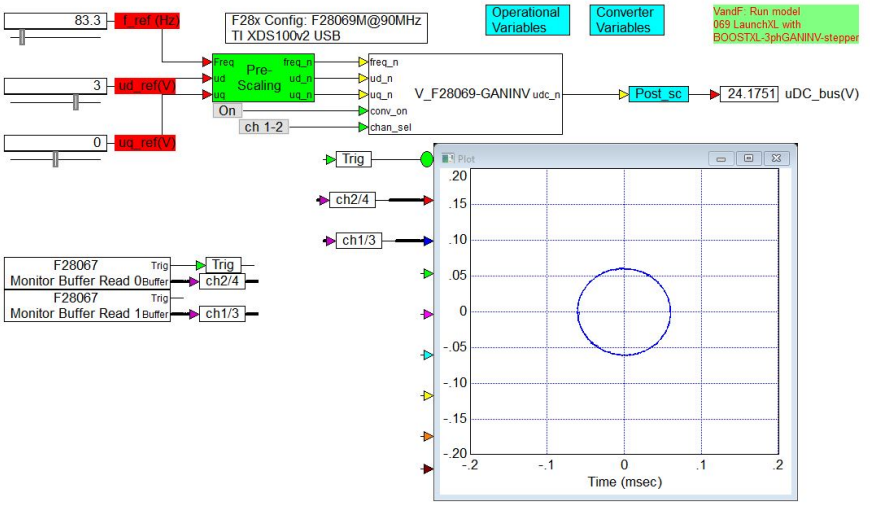


Fig. 5.16 HIL-Debug model, F069-GANINV of a hybrid stepper drive with voltage & frequency control

measured voltage vector locus. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus. Note that the voltage reference slider `ud_ref` is set to 3.0 V (and `uq_ref`= 0), hence the per unit voltage amplitude should be $|\vec{u}|_{\text{ref}} = {}^{3.0}/_{48} = 0.062$, given that a full-scale voltage of u_{fs}=48 V is used.
Observation of the plot result confirms that the amplitude is approximately equal to the anticipated value. Some amplitude deviation will inevitably occur due to practical issues such as: converter dead-time and voltage drop across the converter switches. The electrical frequency reference in figure 5.16 is set to 83.3 Hz, which corresponds to a shaft speed equal to $n_m = {}^{60\, f_{ref}}/_{p} = 100$ rpm, given that a fifty pole pair machine is used, hence $p = 50$.

5.2 Sensorless Field Oriented Control of Hybrid Steppers

The term 'field-oriented control' (FOC) implies that a synchronous 'd,q' coordinate system is tied to the PM flux vector of the hybrid stepper. More specifically, the 'd-axis' of said coordinate system is coincidental with the flux vector $\vec{\psi}$, as may be observed from figure 5.17.
The term 'sensorless' FOC, implies the omission of a mechanical rotor position sensor, typically used to align the controller d-axis with the magnetic flux vector. For 'encoderless' (sensorless) field oriented control, use is made of an

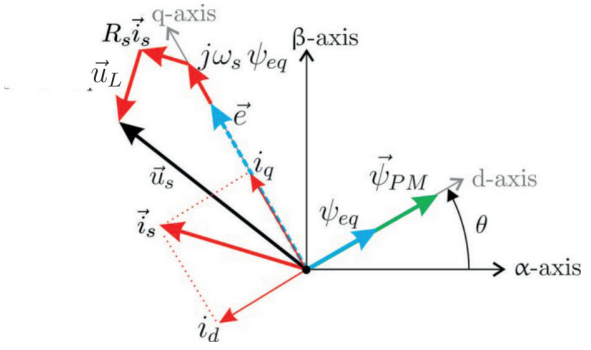


Fig. 5.17 Vector diagram: sensorless field-oriented control (FOC)

algorithm, such as FAST®, to identify the magnetic flux vector (amplitude ψ_{PM} and orientation θ in α, β coordinates) of the hybrid stepper. Correspondingly, the 'd-axis' is then aligned with the estimated magnetic flux vector of said machine.

For hybrid steppers the process of flux estimation is also (as done for IM, PMsyn and IPM) tied to EMF identification, which is based on measurement of the terminal voltage vector $\vec{u}_s$ and current vector $\vec{i}_s$. This EMF identification can be done in two ways, as discussed for IPM (see section 9.2). Note that the use of the IPM approach for hybrid steppers, stems from the fact that this type of PM machine has saliency, which will affect operation of the drive when a non-zero direct axis current is applied, e.g for field-weakening.

The two options for EMF identification are briefly discussed here with aid of figure 5.17 and are as follows:

- Option 1: Use of the voltage $\vec{u}_L$ across the quadrature inductance L_q. This approach identifies a resultant EMF due to PM flux AND the equivalent flux $\psi_{\mathrm{eq}} = i_d \Delta L$ hence, $\vec{e}_{\mathrm{tot}} = \vec{e} + \vec{e}_{\mathrm{eq}}$ (which are both aligned with the 'q-axis'), with $\vec{e}_{\mathrm{eq}} = j\omega_s \psi_{\mathrm{eq}}$. Identification of the EMF vector $\vec{e}_{\mathrm{tot}}$ is then carried out using:

$$\vec{e}_{\mathrm{tot}} = \vec{u}_s - \vec{u}_L - R_s\vec{i}_s \tag{5.2}$$

The advantage of this approach is that it requires knowledge of the voltage across the stator resistance and (only) the quadrature axis inductance defined as: $\vec{u}_L = j\omega_s L_q \vec{i}_s = j\left(\vec{i}_s / i_{\mathrm{sc}}\right) e$, where $i_{\mathrm{sc}} = \psi_{\mathrm{PM}}/L_q$ is the short-circuit current of the hybrid stepper. Furthermore, torque estimation is

simplified, as a sensorless algorithm (that uses this option) identifies the resultant flux $\psi_{\text{tot}} = \psi_{\text{PM}} + \psi_{\text{eq}}$, which means that a shaft torque estimate can be found using $T_{\text{me}} = p\, i_q\, \psi_{tot}$ (note that for two-phase machines, the factor $3/2$ used for three-phase machines must be omitted).

- Option 2: Use of the voltage across the quadrature inductance L_q AND the incremental inductance $\Delta L = L_d - L_q$.
  This approach identifies the EMF due to PM flux hence, $\vec{e}$ (which is aligned with the 'q-axis') with $\vec{e} = j\omega_s \psi_{\text{PM}}$. Identification of the EMF vector $\vec{e}$ is then carried out using:

$$\vec{e} = \vec{u}_s - \vec{u}_L - R_s \vec{i}_s - j\omega_s \psi_{\text{eq}} \tag{5.3}$$

  This approach requires knowledge of the voltage across the stator resistance, the quadrature axis inductance $\vec{u}_L = j\omega_s L_q \vec{i}_s$ and the voltage across the incremental inductance $j\omega_s \psi_{\text{eq}}$. Hence, this approach requires knowledge of the direct and quadrature inductances. Furthermore, torque estimation is more complex, as a sensorless algorithm (that uses this option) identifies the PM flux ψ_{PM}, which means that a shaft torque estimate must now be found using $T_{\text{me}} = p\, i_q\, (\psi_{PM} + i_d \Delta L)$ (again the factor $3/2$, as required for three-phase machines, is omitted here).

The presence of saliency in a hybrid stepper, can be readily verified by operating the drive using 'option 1' and adding a direct axis current component (as will be discussed in the HIL subsection). The effect will be that an equivalent flux component ψ_{eq}, is generated and this will be immediately obvious as the total flux estimated by, for example FAST®, will change accordingly. If the incremental inductance is positive, as is the case in the following hybrid stepper motor example, a positive direct axis current component will INCREASE the total flux ψ_{tot}.
In the following hybrid stepper example, 'option 1' is used, which means that direct axis current variations will affect the flux estimate (as will be discussed), when dealing with a non-salient PM machine, like the hybrid stepper considered in this section.

5.2.1 Sensorless Field Oriented Control of hybrid stepper using FAST®-TI

The example linked to Texas Instruments will be covered using the development toolchain component:PIL→ HIL based on the use of the Piccolo [23] series and F28069 Launchpad. This MCU has access to the Texas Instruments software InstaSPIN®, which contains their classified sensorless algorithm FAST®. Access to FAST®is by way of an Embed DLL, which implies that simulation is not an option when using this algorithm. Initially the tool-chain component presented in the following subsection will be based on the use of the F28069-GANINV drive setup with a Hybrid stepper [11].

5.2.1.1 FAST FOC-Stepper-F069-GANINV:PIL

Reference files:

- `FOC_ST_FAST_F069-GANINV-PIL.vsm`
- `ST-FOC_FAST-PIL_mod.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F28069M, which contains the DLL used to communicate with FAST. This implies the use of a target execution module for file, `ST-FOC_FAST-PIL.out`, as indicated in figure 5.18.
Observation of figure 5.18 shows the presence of an IRTF based two-phase hybrid stepper module, which is connected to a load module that generates a quadratic torque speed curve, which produces 90 mNm at 1000 rpm. An 'average voltage per sample' `AVS conv` module acts as the converter, as discussed earlier (see section 4.1.1.1). The three low-pass filters model the actual R-C filters present in the converter, as used to remove the PWM switching component. In this case (given the use of a two phase machine) the LPF voltages `ualpF`, `ubetF` are found using two `sum` units, hence no Forward Clarke conversion is needed in this case, because we are dealing with the control of a two-phase machine. This is very convenient, given that all motor control is executed using a two-phase approach, i.e in stationary α, β coordinates or direct/quadrature axis d, q coordinates.
Note that the direct and quadrature axis inductance values used by FAST®(see figure 5.18) are purposely set equal to the quadrature axis inductance value of this machine (see the dialog box shown in figure 5.7). The reason for this is that 'option 1', as discussed earlier (see subsection 5.2), is applied here for FOC sensorless control of a salient PM machine (which is a hybrid motor stepper in this case).

The target interface module that executes the out-file, generates the modu-

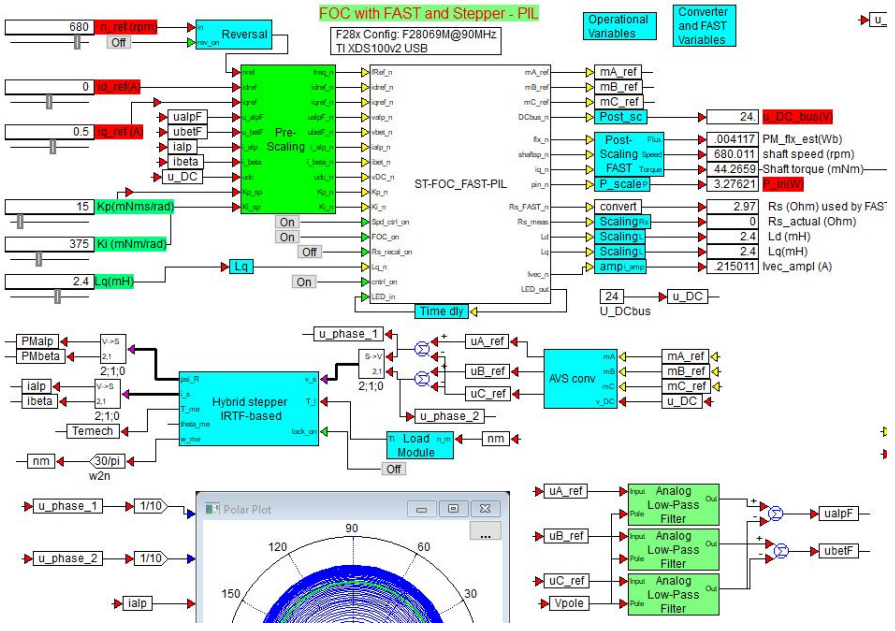


Fig. 5.18 Field-oriented torque/speed control of a hybrid stepper using FAST®:PIL operation

lation indices required for the converter/motor, and these components are not affected by use of the PIL. Generation of the out file is based on the PIL module (see figure 5.19), which must be compiled using the command: `toolbox>codegen`.

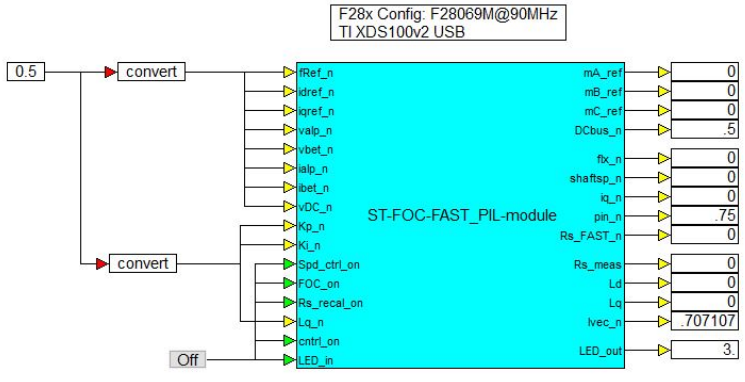


Fig. 5.19 Module:'ST-FOC-FAST-PIL'

Moving one level into the PIL module `ST-FOC-FAST-PIL` (see figure 5.20)

shows a 'Drive Controller'.

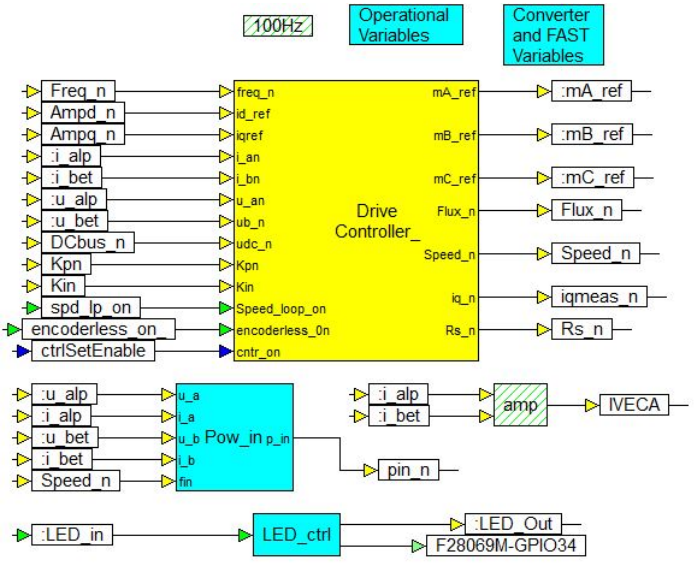


Fig. 5.20 Content PIL module, Field-oriented torque/speed control of a hybrid stepper with FAST®

Added to said controller is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that the PIL control is operational. Furthermore, the modules responsible for input power measurement `Pow_in` and current vector amplitude `amp` are located outside the 'Drive Controller'.

Moving one level down into the `Drive Controller_` module, leads to a set of modules, given in figure 5.21. The `Speed Controller` was introduced earlier, see section 4.3.1.1.

The `Current controller` deployed here, is different (when compared to three-phase examples discussed earlier), given the use of a two-phase machine, as may be observed from figure 5.22.
The measured phase currents from the stepper `i_alp`, `i_bet` are Forward Park converted to (d, q) format. The modulation outputs for the reverse Park module are now used by the `SVM 2-ph` module introduced earlier (see figure 5.5). The current controller requires a per unit inverse bus voltage input, shown as variable `Iudc_n`, which is now generated by the FAST®module `InstaSPIN FOC`. Said unit provides per unit estimates of the rotorflux amplitude `ROTORFLX`, rotor flux angle `ang`, and electrical frequency `rotfreq`. Inputs to the FAST®module are the per unit measured LPF phase voltages and currents, as mentioned earlier. In addition, a variable `CurCTLstate` (gen-

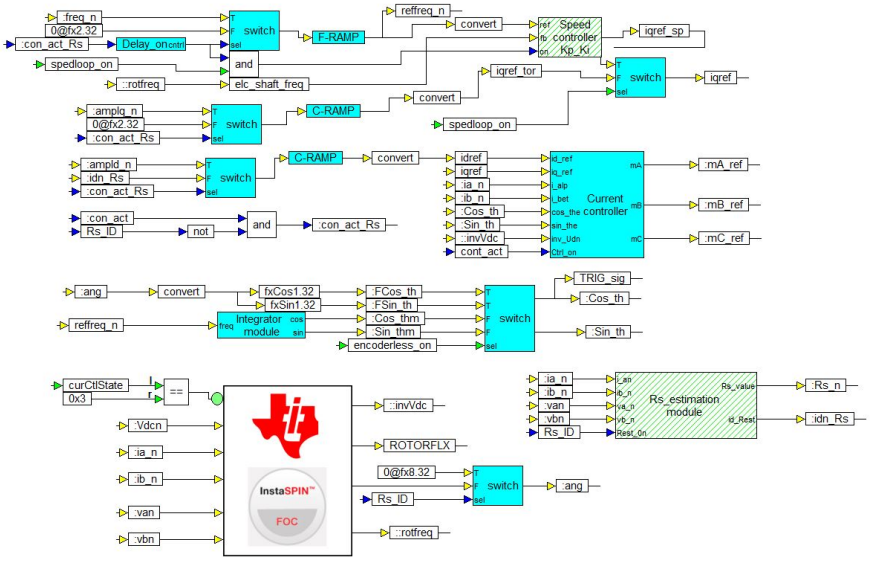


Fig. 5.21 Content 'Drive Controller' module

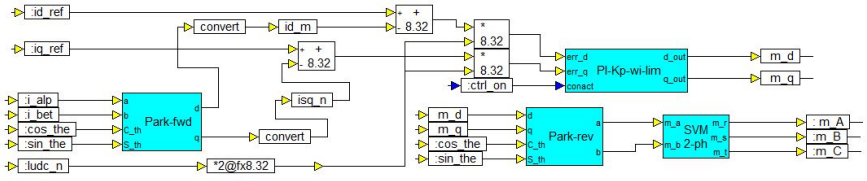


Fig. 5.22 Content 'Current Controller' module

erated in the `100 Hz` module) activates the FAST®module, which is also provided with the per unit measured DC bus voltage `Vdcn`. Also shown in figure 5.21 is a module `Rs_estimation`, which (when activated by the user with variable `Rs_ID`) estimates the stator resistance by injecting a direct axis current (amplitude set in the 'Operational variables' dialogue box). When stator estimation is active the angle output of the FAST®module is held to zero.
When FAST®is not used (encoderless off) the logic variable `encoderless_on` (see figure 5.21) is zero and this implies open loop current&frequency control is active. However, the FAST®module will also remain active under these conditions, hence an effective mode to test operation of this module.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational.

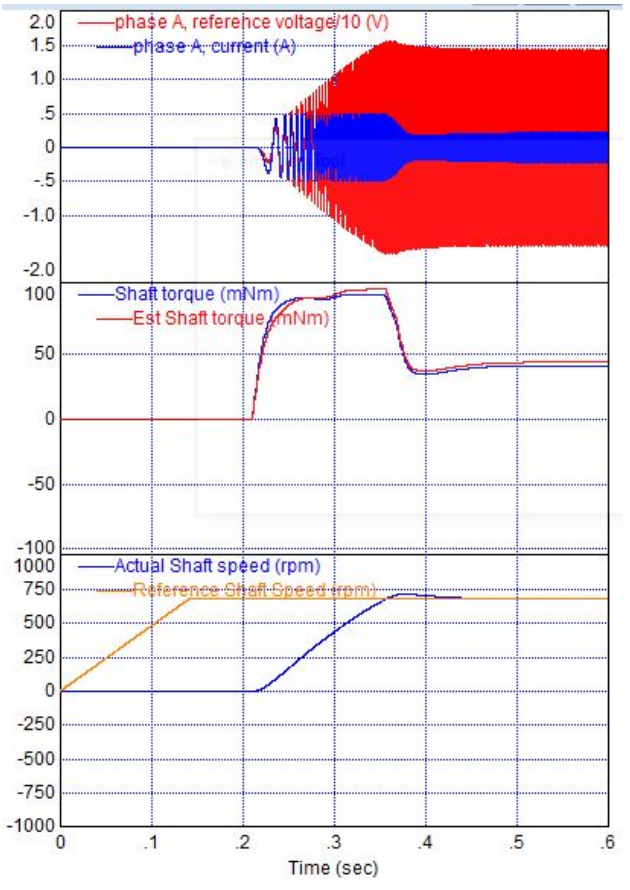


Fig. 5.23 PIL Result: FOC sensorless operation of a hybrid stepper using FAST® with speed control

Operation of the drive is set with a reference speed of 680 rpm and the load controls the quadrature current amplitude (once steady state operation is achieved), as mentioned earlier. FAST® operation starts at 0.22 s and the torque is initially limited by the 0.5 A quadrature current setting for the speed controller, as may be observed from figure 5.23.
The vector plot (see figure 5.24), corresponds to time mark 0.6 s, and this is important because it shows the scaled (by a factor 333.3) rotor flux vector ('green'), current vector ('red'), and scaled (by a factor 1/10) terminal voltage vector ('blue').
The key observation from this vector plot is that the rotor flux vector and current vector (which is due the quadrature current) are orthogonal as required

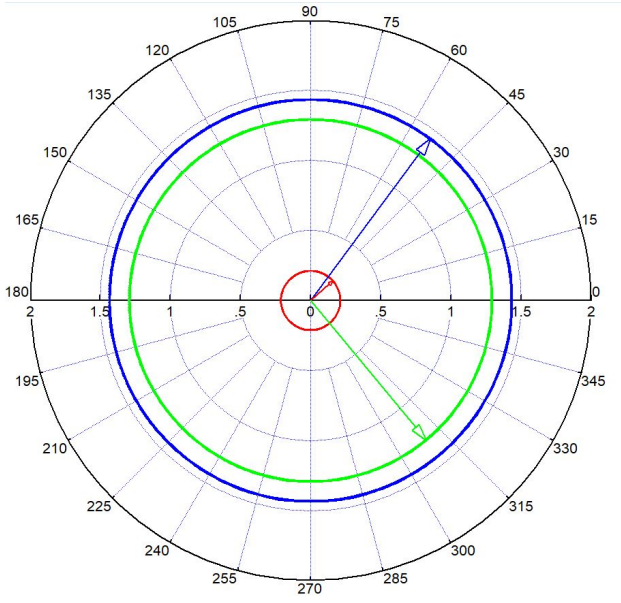


Fig. 5.24 PIL result, hybrid stepper operation, sensorless FOC-speed control active: vector plot

and expected. Furthermore, the voltage vector amplitude at the steady state shaft speed of 680 rpm is 14.4 V, which corresponds to a modulation index of 1.2 , which is more than the theoretical 1.15 limit of three-phase drives. This is attributed to the use of a three-phase converter with a two-phase motor.

5.2.1.2 FAST FOC-Stepper-F069-GANINV:HIL

Reference files:

- `Stepper-FAST_F069-GANinv-compile.vsm`, for HIL-compile
- `Stepper-FAST_F069-GANinv-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), shown in figure 5.25, represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F28069` module), which is connected to a hybrid stepper motor[11]. As mentioned earlier, connections of the two phases are important and in this figure terminal `VC` must be connected to both phases, as shown in figure 5.1.

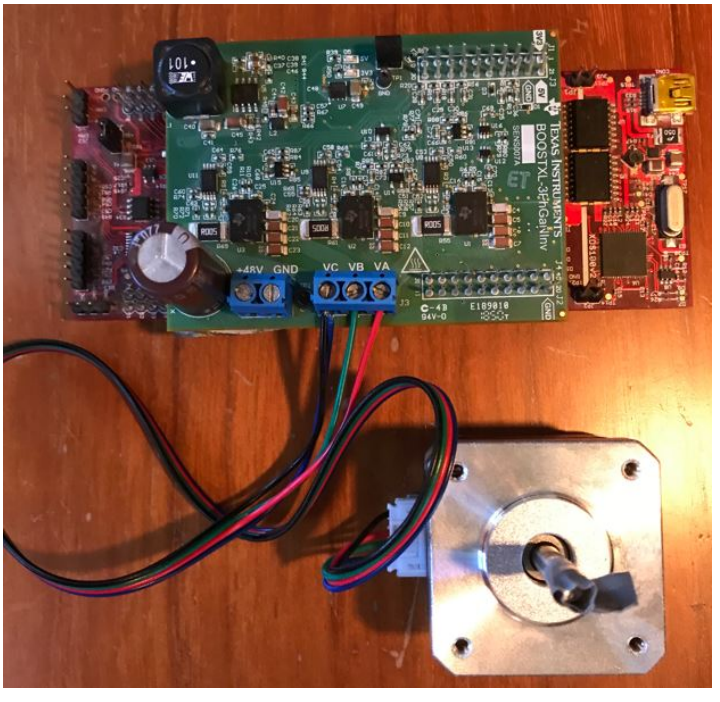


Fig. 5.25 Hybrid stepper drive for HIL operation

For HIL operation, the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches.
A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given

controller configuration. The model given in figure 5.26 shows the `FOC_FAST` module, which must be compiled to generate an out-file.

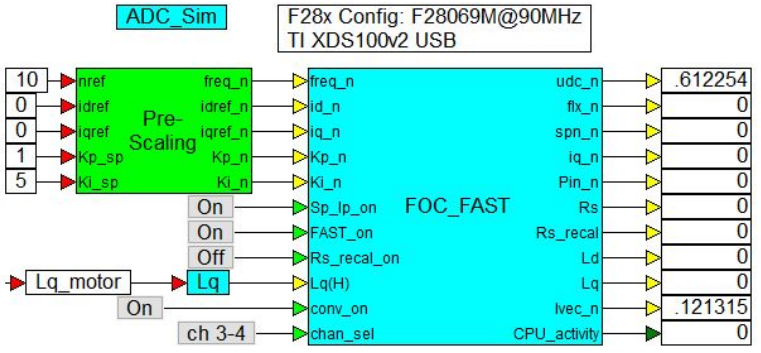


Fig. 5.26 HIL-Compile model of a hybrid stepper drive with field-oriented sensorless control using FAST®

Inputs, shown in this stage as constants (for readability purposes), for this controller module are the user direct and quadrature axis reference currents `id`, `iq` and the reference shaft speed `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller. Also added to this example is an input for tuning/setting the quadrature axis inductance used by FAST®. Five logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed.
These define the following functions:

- `Sp_lp_on`: Speed loop control, when active the shaft speed slider controls the speed, based on the estimated shaft speed generated via FAST®.
- `FAST_on`: Activates sensorless FOC using FAST®. Note when OFF the FAST®algorithm is still operational and this is useful when C&F control is active (both `Sp_lp_on` and `FAST_on` buttons: OFF).
- `Rs_recal_on`: Measures the stator resistance at start-up (when `conv_on` is ON) by injection of a direct axis current (value set in the 'operational variables' dialogue box). The duration of the stator resistance estimation time is set in the `100Hz` module via a time delay setting. It is important to verify that the estimation time is sufficient to reach a steady state resistance estimate.
- `conv_on`: Activates the controller.
- `chan_sel`: Determines which two measurements are shown on the plot unit.

Outputs of the `FOC_FAST` module are:

- `udc_n`: The per unit DC bus voltage, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- `flx_n`: The flux estimate generated by FAST®, which in this case is the sum of the PM flux and equivalent flux ψ_{eq}, as EMF estimation is based on the use of 'option 1', tied to expression (5.2).
- `sp_n`: A per unit shaft speed estimate `sp_n` generated via FAST®.
- `iq_n`: The per unit measured quadrature current, which makes use of the estimated flux, in this case the sum of the equivalent flux $\psi_{eq} = i_d\,(L_d - L_q)$, PM flux and pole pair number to calculate the shaft torque.
- `pin_n`: A per unit estimate of the motor input power. This variable is also useful as an estimate for the drive input power (but this ignores the converter losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx {}^{p_{\mathrm{in}}}/_{u_{\mathrm{DC}}}$.
- `Rs`: The per unit phase resistance value used by FAST®.
- `Rs_recal`: The per unit stator resistance value generated by the resistance estimator at start-up (when `Rs_recal_on` is ON). When selected the estimated stator resistance value is transferred to the FAST algorithm on completion of the estimation sequence. When not active the value shown remains zero and is not used by FAST®.
- `Ld`: The direct axis inductance used by FAST®, which is equal to the sum of the quadrature inductance (set by the user slider) and incremental inductance ΔL (set in the dialog box). In this case, use is made op EMF estimation based on 'option 1' (see subsection 5.2), which implies that ΔL must be set to ZERO.
- `Lq`: The quadrature inductance used by FAST®, controlled by the slider input.
- `ivec_n`: The measured per unit current vector amplitude.
- `CPU_activity`: Provides the user with a % reading of MCU activity, relative to the controller sampling time.

Also shown in figure 5.26 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the `FOC_FAST` module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes (see figure 5.27).
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 5.28), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current(s)-frequency. In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the speed controller sampling frequency is typically set lower then used by

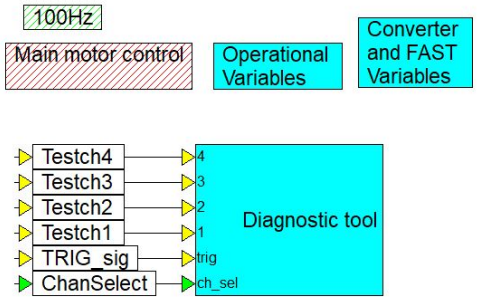


Fig. 5.27 One level down into the FOC-FAST module

the main controller. In this case it is set to 1 kHz. These setting are similar to those introduced for FOC with encoder, as discussed in section 4.3.1.4. Added to this dialogue box is the variable `Res_est_current`, which is the DC injection current value used for stator resistance estimation when the option `Rs_recal` is selected. Note that the modulation index used here, is much higher then theoretically possible for three-phase drives. This is due to the use of modulator that allows a two phase machine to be used with a three-phase converter, as discussed earlier (see section 4.1.1.1).

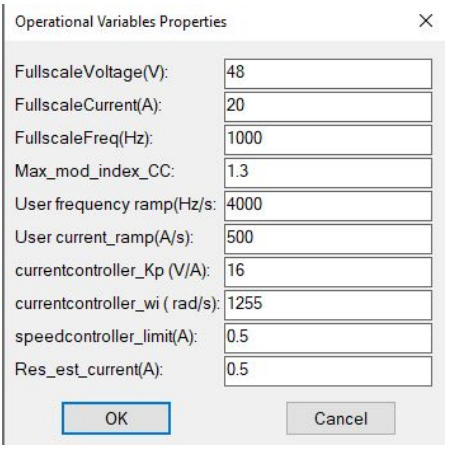


Fig. 5.28 HIL: Dialogue box 'Operational Variables'

The 'Converter and FAST Variables' dialogue box (see figure 5.29), defines the scaling used by the Analogue to Digital converter, as discussed earlier (see equation (4.2)).
This module also shows the low-pass filter corner frequency used by the converter for measuring the half-bridge voltages (see figure 4.12), and is an

important parameter for FAST®, as it is used to compensate for speed dependant phase angle and amplitude errors of the LPF terminal voltage vector.

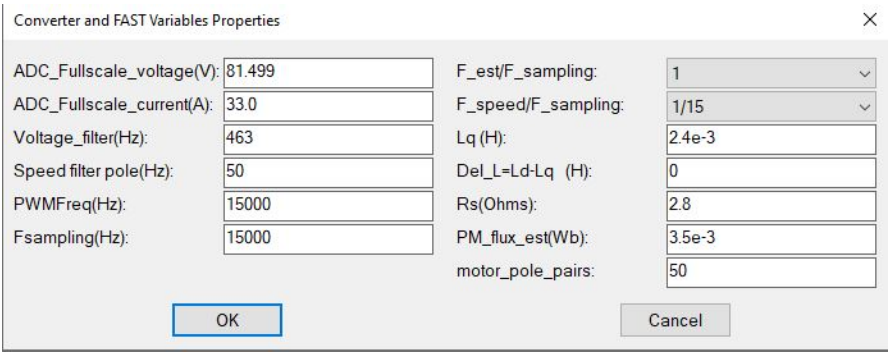


Fig. 5.29 HIL: Dialogue box 'Converter and FAST Variables'

Also added to this dialogue module are a number of FAST®parameters namely:

- `F_est/F_sampling`: set the sampling rate used by FAST®relative to the controller sampling rate, typically set to 1.
- `PM_flux)est (Wb)`: an important parameter which represents an estimate for the PM rotor flux in Wb. This value can be estimated, by using the drive in the C&F mode, see the 'Post Analysis' 4.2.1.5). Typically this value should be less or equal to the PM flux value of the machine.
- `Speed filter pole`: corner frequency of the first order LP filter used by the FAST®speed controller.
- `Rs (Ohms)` the stator resistance value which is an important FAST®variable at start-up of the drive. In this context the EMF threshold speed has been introduced (see equation (2.12)), which for this drive is equal to 72 rpm, when operating at rated torque of 97 mNm (set by the speed controller quadrature current).
- `L_q`: the quadrature inductance, which can be estimated using an L-C-R meter (as required to set the current controller gain). The initial value used here, is replaced by the slider value when FAST®is operational.
- `Ld-L_q`: the difference between the direct and quadrature inductance. This value must be set to zero , when using EMF 'option 1' estimation, despite the fact that a salient PM machine is used (see subsection 5.2).
- `p`: pole pair number, which can be found by operating the current in C&F mode and observing the rotational speed of the rotor when, for example setting the reference shaft speed to 60 rpm (one rotation per second).

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables, `Test1`

and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active. Furthermore, this unit generates a number of FAST®settings, which can generally be left unchanged.

The key component shown in figure 5.27 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit.
Moving one level lower into the 'Main motor Control' module reveals a set of modules (see figure 5.30). Of these, the 'Drive controller' has been discussed earlier (see figure 5.21), where it was used in phase: PIL of the development tool-chain.

The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module.
This module is configured with the following basic settings :

- PWM frequency: 15 kHz
- ADC sampling rate: 15k kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F28069` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present, which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.
The `ADC-PWM-LaunchXL-069M-3phGANINV-2ph` unit is used generate the PWM signals for the converter. In addition it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using two `SUM` modules, as discussed earlier (see figure 5.15). Two per unit phase currents are measured in α, β coordinates, with outputs `i_alp`, `i_bet` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the `100Hz` module) and controlled by said module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example (see figure 5.31), and its task is to execute the `XXX.out` file

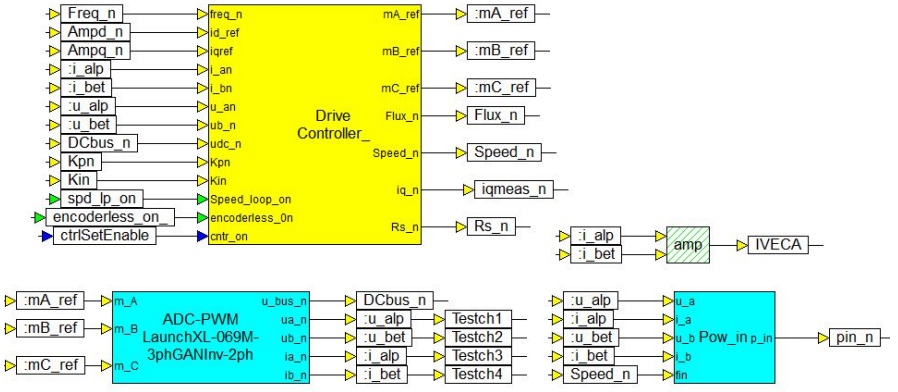


Fig. 5.30 One level down into the 'Main motor controller' module

compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. Three sliders are used to set the direct axis/quadrature axis current reference values and the shaft speed. Three additional sliders are present, of which two are used to se the proportional and integral gain of the speed controller. The third slider, is used to set the quadrature axis inductance value, as required for FAST®.
Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Furthermore, a set of numerical display units have been added which show: estimated equivalent flux, shaft speed, shaft torque, stator resistance used by FAST®, measured stator resistance at startup (when this function is actively used), direct and quadrature axis inductance values used by FAST®and current vector amplitude. In addition, a set of gauges have been added which show (low-pass filtered): shaft speed, shaft torque, input power and estimated DC supply current.
Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.
The plot given in figure 5.31 displays the per unit measured LPF vector locus. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus. The actual amplitude of the voltage locus is found by by using the 'Read Coordinates' function of the plot module, which shows a value of 0.177 , that must be multiplied by the 48 V full scale voltage which gives a value of 8.5 V. Note that this value is not in agreement with the PIL result (see figure 5.24). The reason is that the shaft speed reference shown in figure 5.31 was set to 680 rpm, which corresponds to an electrical frequency f_s of 566.6 Hz. The corner frequency f_c of the LPF is 463 Hz, hence the true amplitude of the voltage locus is found by multiplying

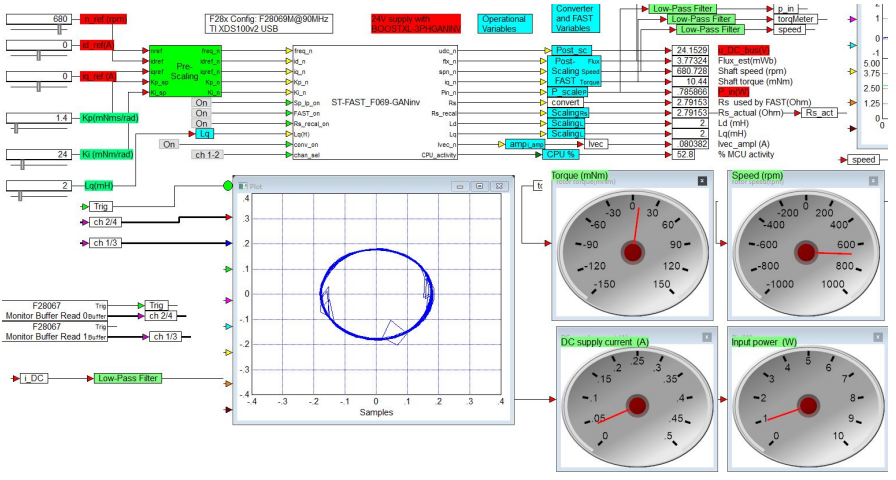


Fig. 5.31 HIL-Debug model, F28069M-GANINV of a hybrid stepper drive with field-oriented control using FAST®

the measured amplitude with a factor $\sqrt{1+(f_s/f_c)^2}$. This gives a true voltage locus amplitude of 13.4 V, which is more in agreement with the PIL result (see figure 5.24). These results show operation under encoderless FOC-speed control with the machine operating under no-load conditions.

Post Analysis: Introduction

In this section an overview of some important tests will be discussed, that are helpful for commissioning a sensorless drive with a hybrid stepper. For this purpose use is made of a time plot figure, as shown below (see figure 5.32). Said plot shows the estimated rotor flux amp normalised to the PM rotor flux value set in the 'Converter and FAST variables' dialogue box (hence, if the value set in said dialog box, matches the estimated value then the normalised flux shown in figure 5.33 will be 1. Furthermore, this plot shows the current vector amplitude and the estimated stator resistance (active when the `Rs_recal` function is used) over a 20 s time frame.

The example given here shows stator resistance estimation at start-up when the `Rs_recal` function is active. Initially, the current amplitude moves to 0.5 A, as expected as this is the value set in the 'operational' dialogue box for Rs estimation. The duration of the Rs estimation is set by the unit `Timer Rs` located in the `100 Hz` module and should be adjusted to ensure that this routine is switched OFF when a satisfactory steady state resistance value has been reached. Note that after termination of the stator resistance estimation the drive moves to sensorless FOC control as evident by the normalised flux curve. This plot also conveys the message that the rotor flux set in the 'Converter and FAST variables' dialogue box is slightly lower then the value estimated by FAST®. Note in this context that the flux estimation is now

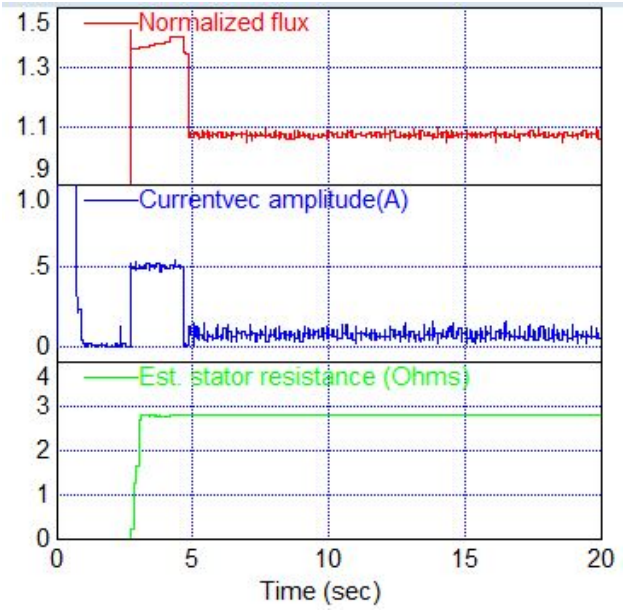


Fig. 5.32 HIL-Debug model, F069-GANINV of a hybrid stepper drive:time plot showing Rs estimation at start-up

the sum of the PM flux and the equivalent flux. The latter will be present if a machine is considered (or suspected) to be of a non-salient nature. This will be examined in the ensuing alina.

Post Analysis: Inductance Estimation
Tuning the quadrature axis inductance is done by first using the `Rs_recal` function to measure the stator resistance. Furthermore, the direct axis reference is purposely set to zero to avoid any equivalent flux component in the estimated flux, which will in this case be the PM flux of the machine. This value should NOT change significantly if (hand) shaft load changes are applied, which vary the quadrature current (FOC operation under speed control active). Tuning of the quadrature inductance (using the slider provided) can be used to minimize the flux changes, as a result of the quadrature axis inductance changes, as may be observed from figure 5.33.
These results show that quadrature axis current changes do not have a significant impact, given a quadrature axis inductance slider setting of 2 mH.

An interesting and important test is to introduce a positive 0.2 A direct axis current step change, which is applied $t \approx 10$ s and removed $t \approx 16.3$ s in this case. Observation of the normalised flux (see figure 5.34), shows that an equivalent flux is introduced (which is added to the PM flux), which is equal to 0.77 mWb for this machine. This implies that a non-zero incremental

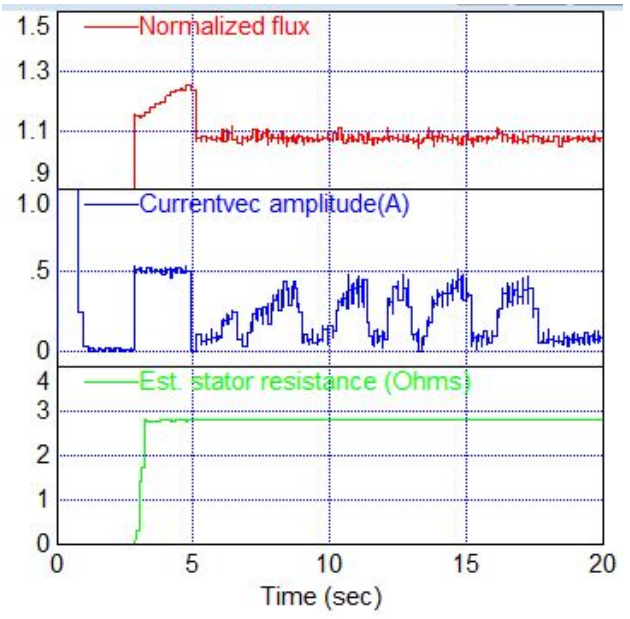


Fig. 5.33 HIL-Debug model, F069-GANINV of a hybrid:time plot showing effect of quadrature axis current changes on normalised flux

inductance for this hybrid stepper is indeed present and equal to 3.85 mH, which corresponds to a direct axis inductance of 5.8 mH (given the quadrature axis inductance value found in the previous test).

Hence, an important observation is that the hybrid stepper, behaves very much like a 'flux enhancement' IPM, as discussed in chapter 9. Consequently, control methods as discussed in said IPM chapter can be applied to the hybrid stepper drive but the equivalent flux contribution to the total flux, is limited hence the reluctance torque contribution will also be limited. Therefore, the so called 'MTPA' control approach, based on expression (9.5), is not applied to this hybrid stepper drive example.

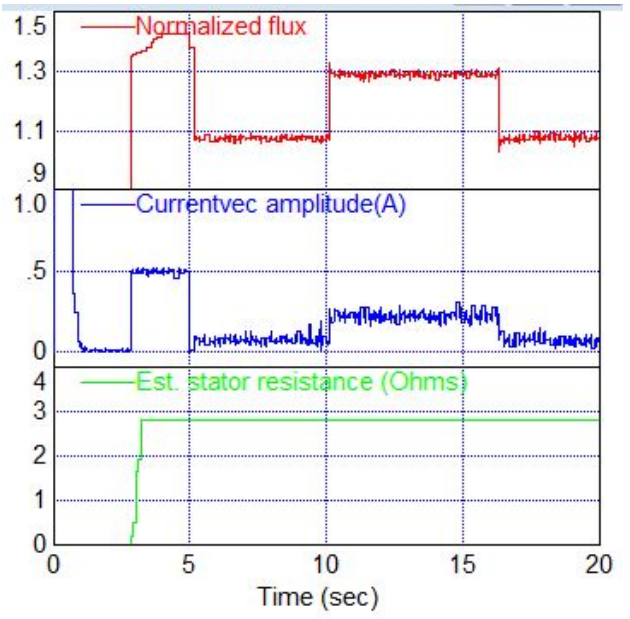


Fig. 5.34 HIL-Debug model, F069-GANINV of a hybrid stepper drive:time plot showing effect of a positive direct axis current change on normalised flux

Post Analysis: FOC performance estimator

Reference file:

- `FOC performance estimator-Stepper.vsm`, for FOC performance estimation.

The performance of a field-oriented drive, such as the hybrid stepper considered here, is dependent on a number of critical parameters. For example the DC bus voltage and maximum modulation index used set a voltage amplitude locus, which can be used to calculate the maximum EMF that is available. This analysis is based on the use of figure 5.17 and also requires knowledge of the direct/current current setting and motor parameters. The reference file: `FOC performance estimator-Stepper.vsm` calculates the FOC speed limit, with inputs/outputs as shown in figure 5.35.

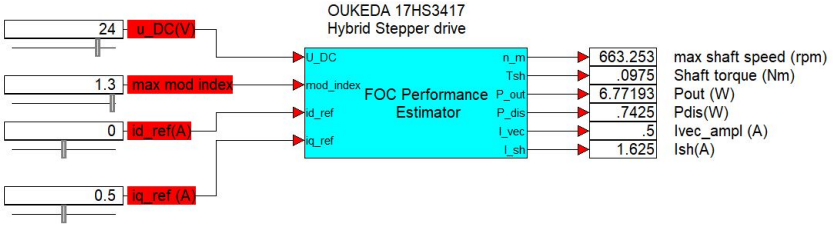


Fig. 5.35 FOC performance estimator of a hybrid stepper drive:operation with $i_q^{\text{ref}} = 0.5(A)$, $i_d^{\text{ref}} = 0(A)$

The dialog box for this drive is identical (with exception of the inertia used) to the one introduced earlier (see figure 5.7) and closely matches the experimental machine used for HIL operation. The slider setting match what is used for operation HIL and the result show the maximum shaft speed that can be achieve for a given quadrature axis current that is equal to the speed controller limit (as set in the dialog box, see figure 5.28).

The impact of introducing a negative direct axis current to increase the speed range of the drive has been discussed extensively for non-salient PM drives (see section 4.5), which refers to so called 'field-weakening'. For hybrid stepper drives, the use of a negative direct axis current is more complex, given the IPM 'flux-enhancement' nature of the machine as is discussed below.
For example, under no-load conditions (`iq_ref`=0), the speed limit is simply set by the equation, `max mod index` times `u_DC`/2, with condition `id_ref`=0, which leads to the outputs shown in figure 5.36.
These results are closely aligned with the HIL example, when operating under

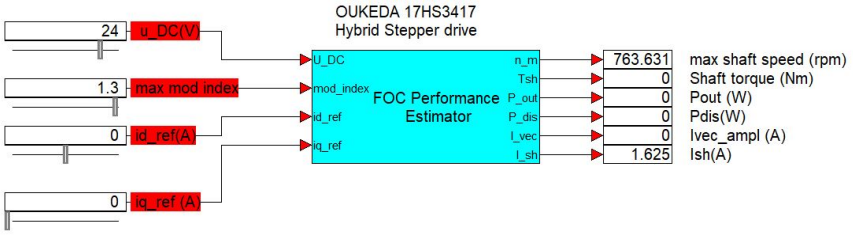


Fig. 5.36 FOC performance estimator of a hybrid stepper drive:operation with $i_q^{\text{ref}} = 0(A)$, $i_d^{\text{ref}} = 0(A)$

the same drive conditions. When adding a negative direct current component to the drive the maximum speed range increases significantly, as may be observed from figure 5.37.

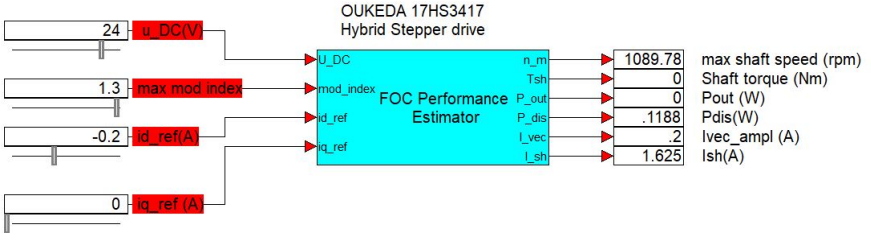


Fig. 5.37 FOC performance estimator of a hybrid stepper drive:operation with $i_q^{\text{ref}} = 0(A)$, $i_d^{\text{ref}} = -0.2(A)$

The reason for the maximum shaft speed increase is two fold namely:

- voltage drop across the quadrature inductance: this value is controlled by the ratio of the current amplitude (0.2 A in this case) and short-circuit current (1.625 A), which provides an increase in the available EMF by a factor $^{0.2}/_{1.625}$, or 12 %. This is precisely how non-salient machine are able to operate at higher speeds.
- reduced total rotor flux : adding a negative direct axis component reduces the total flux by $i_d^{\text{ref}}(L_d - L_q) = 0.7\,\text{mWb}$, hence a factor $^{0.7}/_{3.9}$, or 18 % when compared against the PM flux of the machine.

Hence, the total speed increase due to these two factors is 30 %, which corresponds to the results shown in figure 5.37. Again running the HIL example gives similar results when using the same conditions. Note that the reduction in total flux will also affect the torque output, as may be observed from figure 5.38. Furthermore, the drive is able to operate with modulation index value that exceed those of a standard three-phase drive. This can be directly contributed to the use of a two -phase machine with a three-phase converter.

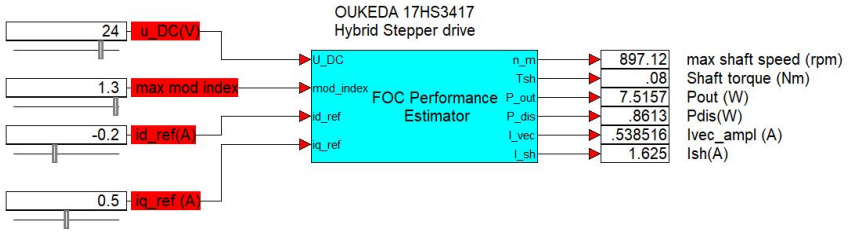


Fig. 5.38 FOC performance estimator of a hybrid stepper drive:operation with $i_q^{\text{ref}} = 0.5(A)$, $i_d^{\text{ref}} = -0.2(A)$

This post-analysis shows that the 'field-weakening' mechanism of hybrid steppers is more elaborate, when compared with non-salient drives. This was to be expected given the IPM type behaviour of the machine.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

Chapter 6
Three-phase Induction Machine Drives

6.1 Voltage-Frequency Control-IM

Purpose of these examples is to realize voltage/frequency and speed control of an induction motor. Use is made on the voltage control approach discussed earlier (see section 4.1). This control approach is then modified to include a voltage/frequency control concept as discussed in [13]. Speed control capability has also been added to some examples, where use is made of the shaft encoder connected to the PM machine, which is (mechanically) connected to the IM under consideration. A V/f controller maintains the ratio of voltage and frequency (in rad/s) at an approximately constant value, given that said ratio represents the flux in the machine. The reference flux for the controller is the rotor flux ψ_R^*, as this largely controls the EMF component, together with the electrical frequency ω_s.
In terms of calculating the required d, q voltage vector components, the voltage across the leakage inductance, defined as $\vec{u}_L = j^{e\,\vec{i}_s}/_{i_{\rm sc}}$ (where $i_{\rm sc}$ represents the equivalent short-circuit current $i_{\rm sc} = \psi_R/L_\sigma$) is ignored, given that the term $(^{e\,\vec{i}_s}/_{i_{\rm sc}})$ is relatively small.
When (approximately) maintaining a constant flux, the shaft torque/speed characteristic $T_m\,(\omega_m)$ is as shown in figure 6.1. Added to said figure is a load torque speed curve $T_l\,(\omega_m)$.

Voltage/frequency Control of the drive is done using the following two methods:

- Control of the reference synchronous speed $\omega_{\rm syn}^*$.
- Control of the reference shaft speed $\omega_{\rm m}^*$, which requires access to a shaft speed sensor or encoder.

When controlling the reference synchronous speed (no speed sensor required) the machine torque/speed curve shifts from operating point A→B and a load torque change ΔT_l will occur (see figure 6.1). Observe that the steady

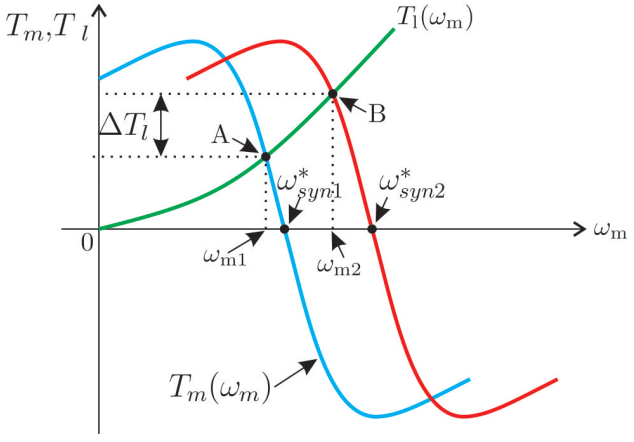


Fig. 6.1 V/f drive operation for induction machine, with reference flux ψ_R^{ref},

state shaft speed will be different from the synchronous speed and in this case changes from $\omega_{\text{m1}} \rightarrow \omega_{\text{m2}}$ (see figure 6.1), where it is noted that the slip frequency $\omega_{\text{slip}} = (\omega_{\text{syn}} - \omega_m)$ increases with load torque. The type of voltage/frequency controller for implementing this approach is shown in figure 6.2.

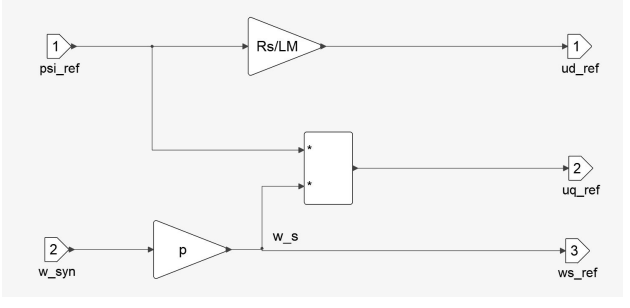


Fig. 6.2 Simplified V/F induction machine control, without speed sensor

The controller representation satisfies the following set of equations [13]:

$$u_d^{\text{ref}} = \psi_R^{\text{ref}} \frac{R_s}{L_M} \tag{6.1a}$$

$$u_q^{\text{ref}} = \omega_s^{\text{ref}} \psi_R^{\text{ref}} \tag{6.1b}$$

$$\omega_s^{\text{ref}} = p\, \omega_{\text{syn}}^{\text{ref}} \tag{6.1c}$$

where L_M, represents the magnetizing inductance of the (four-parameter) machine (as discussed in section 2.1.3.5) and ψ_R^{ref} represents the rotor flux reference variable (Wb). The input synchronous reference shaft speed $\omega_{\text{syn}}^{\text{ref}}$ is multiplied by the pole pair number p, as required to generate the electrical frequency ω_s^{ref}.

The issue with controlling the synchronous reference speed is that the shaft speed will be slip speed dependant. Observation of figure 6.1 shows that when moving from operating point A →B the slip frequency has increased and the shaft speed is not controlled directly. The alternative is to use a V/F controller with shaft speed measurement, as indicated in figure 6.3. In this case the controller calculates the synchronous electrical frequency ω_s^{ref} (and corresponding synchronous shaft frequency $\omega_{\text{syn}} = {}^{\omega_s^{\text{ref}}}/_p$) needed to achieve the (steady-state) condition $\omega_m = \omega_m^{\text{ref}}$.

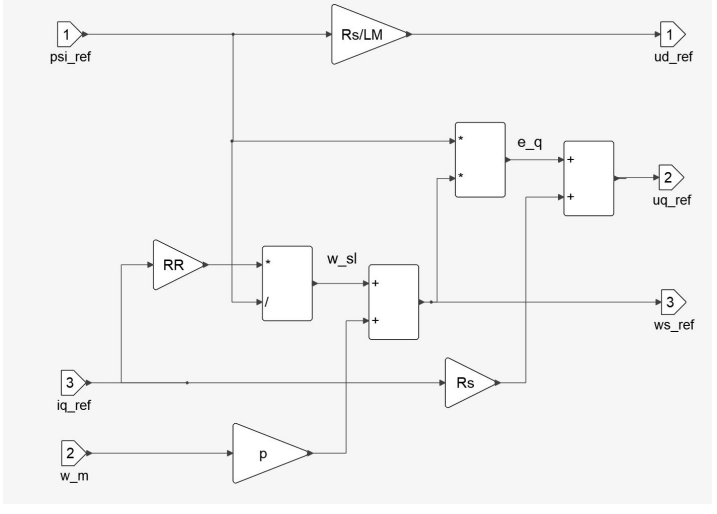


Fig. 6.3 V/f drive for induction machine, with reference flux ψ_R^{ref}, quadrature current i_q^{ref} inputs, that makes use of measured shaft speed ω_{me}

In this case a speed controller compares the reference speed with actual speed and generates a quadrature current reference value. The latter is used to generate the slip frequency that must be added to the electrical shaft speed. Inputs to the controller are the reference rotor flux `psi_ref`=ψ_R^{ref} Wb, measured shaft speed `w_m`=ω_{me} rad/s and reference quadrature current `iq_ref`=i_q^{ref}. Outputs are the reference direct/quadrature axis voltages

$u_d^{\text{ref}}, u_q^{\text{ref}}$ and electrical frequency ω_s^{ref}. The controller representation satisfies the following set of equations [13]:

$$u_d^{\text{ref}} = \psi_R^{\text{ref}} \frac{R_s}{L_M} \tag{6.2a}$$

$$u_q^{\text{ref}} = \omega_s^{\text{ref}} \psi_R^{\text{ref}} + i_q^{\text{ref}} R_s \tag{6.2b}$$

$$\omega_s^{\text{ref}} = p\,\omega_{\text{me}} + \frac{i_q^{\text{ref}} R_R}{\psi_R^{\text{ref}}} \tag{6.2c}$$

The quadrature reference current i_q^{ref} follows from the torque expression

$$T_{\text{me}} = \frac{3}{2}\, p\, i_q^{\text{ref}}\, \psi_R^{\text{ref}} \tag{6.3}$$

where p represents the number of pole pairs of the machine and T_{me} the mechanical shaft torque. The speed controller, with gains K_p, K_i can in the continuous time domain be represented as

$$T_{\text{me}} = \left(K_{\text{p}} + \frac{K_{\text{i}}}{s}\right)\left(\omega_{\text{me}}^{\text{ref}} - \omega_{\text{me}}\right) \tag{6.4}$$

where ω_{me} is the actual shaft speed of the machine (rad/s). Use of said equation in combination with equations (6.3),(6.4) yields an explicit expression for the reference quadrature current i_q^{ref} namely

$$i_q^{\text{ref}} = \underbrace{\frac{2}{3\,p^2 \psi_R^{\text{ref}}}}_{K_{\text{pi}}} \left(K_p + \frac{K_i}{s}\right)\left(\omega_{\text{m}}^{\text{ref}} - \omega_{\text{m}}\right) \tag{6.5}$$

where a gain parameter, K_{pi} is introduced, which will be subsequently used for fixed point controller implementation. Expression (6.5) defines the quadrature current input for the V/f controller shown in figure 6.3 based on the actual (from an encoder in this case) and reference electrical shaft speed ω_m^{ref}. This expression together with equation (6.2) must be scaled using the approach set out in section 1.4.2, which after some manipulation gives:

$$u_{\mathrm{dn}}^{\mathrm{ref}} = \left(\frac{R_{\mathrm{s}}\,\psi_{\mathrm{fs}}}{L_M\,u_{\mathrm{fs}}}\right)\psi_{\mathrm{Rn}}^{\mathrm{ref}} \tag{6.6a}$$

$$u_{\mathrm{qn}}^{\mathrm{ref}} = \left(\frac{2\pi\,\psi_{\mathrm{fs}}}{u_{\mathrm{fs}}}\right) f_{\mathrm{Rn}}^{\mathrm{ref}}\,\psi_{\mathrm{sn}}^{\mathrm{ref}} + \left(\frac{R_s\,i_{\mathrm{fs}}}{u_{\mathrm{fs}}}\right) i_{\mathrm{qn}}^{\mathrm{ref}} \tag{6.6b}$$

$$f_{\mathrm{sn}}^{\mathrm{ref}} \cong f_{\mathrm{en}} + \left(\frac{R_R\,i_{\mathrm{fs}}}{2\pi\,\psi_{\mathrm{fs}}}\right)\frac{i_{\mathrm{qn}}^{\mathrm{ref}}}{\psi_{\mathrm{Rn}}^{\mathrm{ref}}} \tag{6.6c}$$

$$i_{\mathrm{qn}}^{\mathrm{ref}}(t_k) \cong \left(K_{\mathrm{pi}}\frac{2\pi f_{\mathrm{fs}}}{i_{\mathrm{fs}}}\right)(K_{\mathrm{p}} + K_{\mathrm{i}}\,T_{\mathrm{sp}})\,\Delta f_n + i_{\mathrm{qnINT}}(t_{k-1}) \tag{6.6d}$$

with $\Delta f_n = f_{\mathrm{en}}^{\mathrm{ref}} - f_{\mathrm{en}}$. The discretization of equation (6.5) is carried using the approach shown for equation (2.6) which leads to equation (6.6d). A variable T_{sp} is introduced in the latter expression, which represents the time step used for the discrete speed controller module.

6.1.1 Voltage-Frequency Control-TI

Purpose of this section is to realize voltage-frequency control of an induction machine connected to a PMsyn machine, of which the encoder is used for shaft speed measurement/control. The tool chain, as introduced in figure 2.45, is used as a template for the ensuing subsections, but in this case the toolchain: SIM→ HIL, will be considered for a specific hardware example. For this purpose the `F0049-GANINV` module is connected to a induction [25] machine. A PMsyn machine [2] provides the encoder for these examples.

6.1.1.1 Voltage-frequency Control-F0049-GANINV:SIM

Reference file: `Vf_F0049_GANINV-SIM.vsm`

Prior to operating the V-f drive, real-time, it is prudent to test the fixed point controller under simulation conditions as indicated in figure 6.4.
A 'Pre-scaling' module is used to convert and scale the floating point, user (slider) variables to fixed point format. Inputs to the V-f controller module are:

- `fref_n`: per unit reference frequency, which is linked to the reference shaft speed n_{me}(rpm) by a gain factor $\left(\frac{p\,n_{\mathrm{me}}^{\mathrm{ref}}}{60\,f_{\mathrm{fs}}}\right)$.

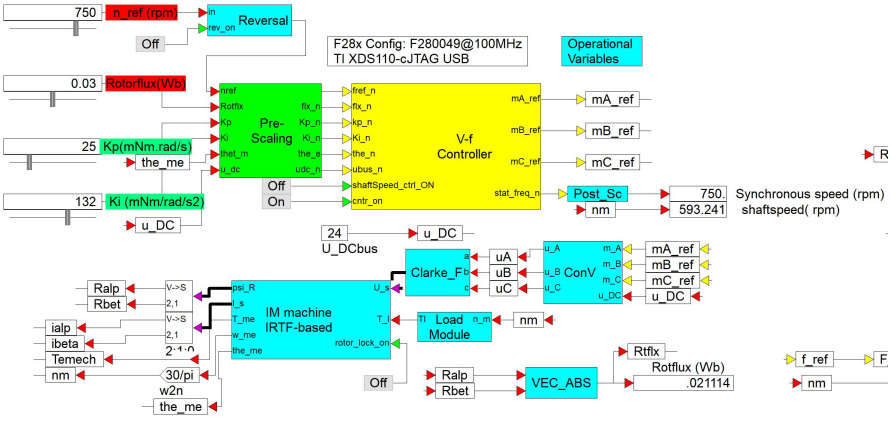


Fig. 6.4 Simulation of a V-f drive with speed control option

- `flx_n`: per unit rotor flux, which is scaled by the full scale flux ψ_{fs} set in the user dialog box (to be discussed).
- `kp_n`: scaled proportional controller speed gain, which is linked to the gain factor K_{pi} as shown in equation (6.6d). The resultant gain factor to be used is therefore equal to $\left(\frac{4\,\pi\,K_p\,f_{\text{fs}}}{3\,p^2\,\psi_R^{\text{ref}}\,i_{\text{fs}}}\right)$.
- `ki_n`: scaled integral controller speed gain. The speed controller gains are in this lab example expressed in terms of the proportional and integral gain as shown by equation (2.3). This implies that a similar gain factor is also applicable here which includes the discrete time step T_{sp} used by the speed controller, which gives $\left(\frac{4\,\pi\,K_i\,T_{\text{sp}}\,f_{\text{fs}}}{3\,p^2\,\psi_R^{\text{ref}}\,i_{\text{fs}}}\right)$.
- `the_n`: per unit electrical shaft angle, which is linked to the encoder shaft angle θ_{me} by the gain factor $(p\,\theta_{\text{me}}/2\,\pi)$.
- `ubus_n`: per unit bus voltage, scaled by the full scale voltage u_{fs} set in the user dialogue box (to be discussed).
- `shaftSpeed ctrl_on`: when active the controller makes use of the measured shaft speed for voltage-frequency control. When `off` the reference speed of the `n_ref` slider represents the synchronous shaft speed.
- `cntr_on`: control active, which enables the controller.

Outputs of the V-f module are the per unit stator frequency `stat_freq_n` together with the phase modulation indices: `mA_ref`, `mB_ref`, `mC_ref`, which are connected to the 'Conv' module. This unit multiplies the modulation indices by half the DC bus voltage and limits the outputs to $\pm\,12V$ prior to use by the induction motor model.
Outputs of the IM model are the currents i_α, i_β, shaft torque T_{me}, shaft speed n_{me}, shaft angle θ_{me} and rotor flux $\psi_{R\alpha}$, $\psi_{R\beta}$. In the HIL phase, the encoder of the PM machine, mechanically connected to the IM motor is used for shaft speed measurement. In addition to the above, the rotor flux amplitude of

the machine is also displayed, given that this variable is controlled by the controller.
The dialogue box 'Operational Variables' (see figure 6.5), contains all the user settings required for this example.

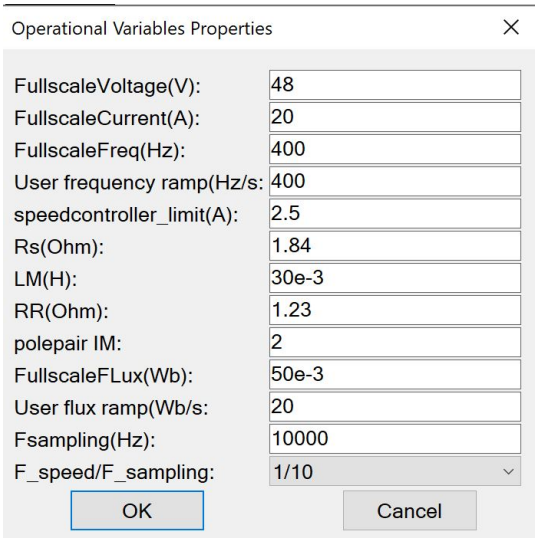


Fig. 6.5 Dialogue box used for drive with a V-f Controller

The dialogue box is used to assign full scale values for the voltage, current, frequency and rotor flux. Dialogue box entries for the maximum allowable rate of change for flux and frequency are also provided, together with a speed controller limit value `speedcontroller_limit(A)`. This value is the maximum (absolute) quadrature current value, which the speed controller is set to provide. Said box also assigns the motor pole pair number and parameters of the induction motor in use. In addition, the sampling frequency used for the V/f and speed controller modules are set by the user in this dialogue box.

Moving one level lower into the 'V-f Controller' module, reveals a set of modules given in figure 6.6. The per unit electrical shaft angle `theta_n` is used to generate a Cosine/Sine variables that are connected to the a '`Speed_est`' module which give a per unit (electrical) shaft speed estimate `shaft_freq_n`.
Inputs to the speed controller are the per unit reference electrical frequency `freq_n` and estimated shaft speed frequency (from the speed estimator) `shaft_freq_n`. Output of the speed controller is the per unit quadrature reference current `iqn`. The reference shaft speed signal is connected to the `ref` input of the Speed controller module via a frequency rate limiter module 'F-RAMP' and switch (with input `con_act2`), which ensures that said

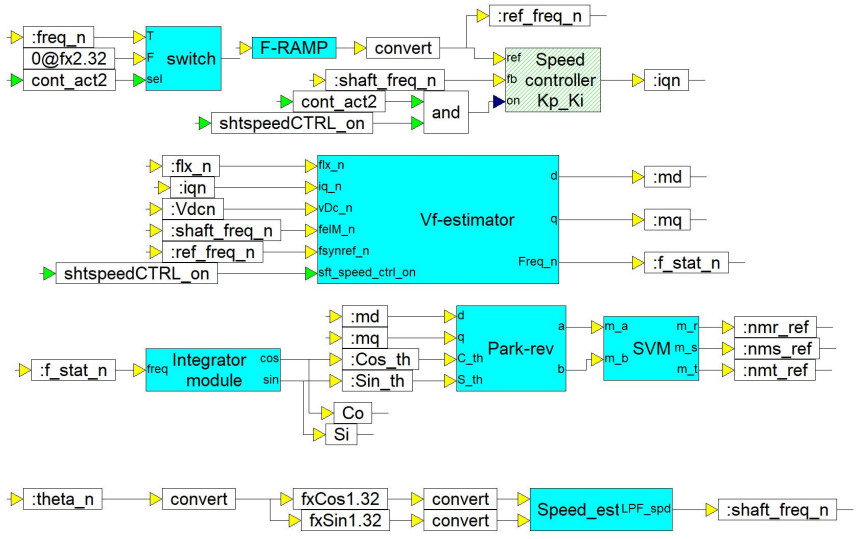


Fig. 6.6 One level into the 'V-f Controller'

variable is held at zero when the converter is disabled. The logic variable `shftspeedCTRL_on` is `on` when speed control using the measured shaft speed is selected. When `off`, the speed controller is disabled (output `iqn`=0) and the 'Vf-estimator' makes use of the variable `ref_freq_n` to control the synchronous shaft speed (see figure 6.2).
As in previous examples a Cosine/Sine function, with variables `Cos_th`, `Sin_th` is used instead of the instantaneous angle of the voltage reference vector. These two variables are then consistently used for subsequent Park transformations, which have been adapted to allow the use of Cosine/Sine instead.
The basic structure of the `Vf-estimator`, as shown in figure 6.7, reflects the control structure of the floating point versions (see figures 6.2, 6.3) where the outputs are given in terms of voltages, not modulation indices, as is the case for the fixed point module. Furthermore, the reference frequency f_s of this unit is scaled by a factor $1/f_{\mathrm{fs}}$.
However, scaling as expressed by inputs to the speed controller, are the per unit reference electrical frequency `ref_freq_n` and estimated shaft speed frequency (from the speed estimator) `shaft_freq_n`. Output of the speed controller, is the per unit quadrature reference current `iqn`. The per unit reference shaft speed signal and the measured shaft speed are connected to inputs `fsynref_n` and `feIM_n` respectively of the Vf-estimator (see figure 6.7).
A brief outline of the variables which appear in the discrete estimator are given below:

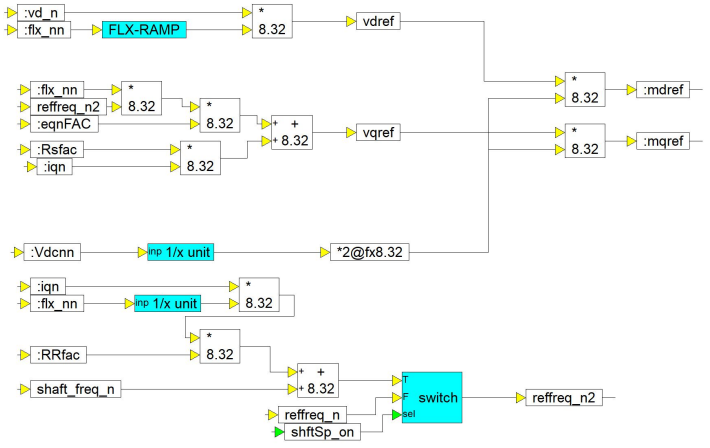


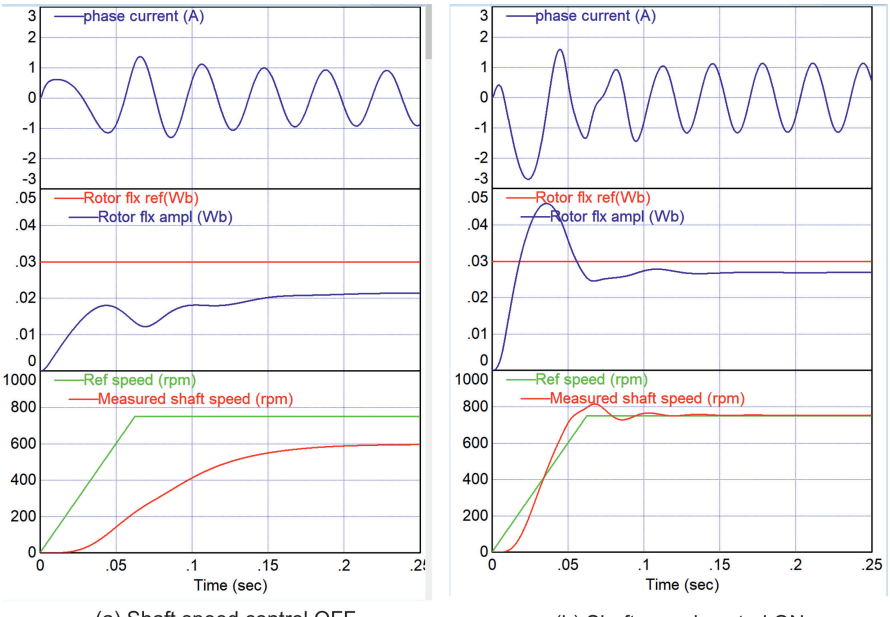
Fig. 6.7 One level into the 'Vf-estimator'

- `vd_n`= $\frac{R_s}{L_s}\left(\frac{\psi_{\text{fs}}}{u_{\text{fs}}}\right)$, where ψ_{fs}, u_{fs} represent the full scale flux and voltage values respectively. Represented by bracketed term in equation (6.6a).
- `flxnn`= $\frac{\psi_R^{\text{ref}}}{\psi_{\text{fs}}}$, which is the per unit rated scaled rotor flux reference value.
- `reffreq_n2`= $\frac{f_s^{\text{ref}}}{f_{\text{fs}}}$, which is the per unit reference stator frequency. This variable is used as an input to the 'integrator' module of the voltage controller.
- `egnFAC`= $\frac{2\pi\psi_{\text{fs}}}{u_{\text{fs}}}$, which is a scaling factor used to scale the term $e_q = \omega_s\,\psi_R^{\text{ref}}$, as represented by the first RHS bracketed term in equation (6.6b).
- `Rsfac`= $\frac{R_s\,i_{\text{fs}}}{u_{\text{fs}}}$, which is the scaled stator resistance, as represented by the second RHS bracketed term in equation (6.6b).
- `iqn`= $\frac{i_q}{i_{\text{fs}}}$, which is the per unit quadrature current generated by the speed controller.
- `Vdcnn`= $\frac{u_{\text{DC}}}{u_{\text{fs}}}$, which is the per unit measured DC bus voltage generated by the ADC/PWM module.
- `RRFAC`= $\frac{R_R\,i_{\text{fs}}}{2\pi\psi_{fs}}$, which is a scaling factor used to scale the term $\omega_r = \frac{i_q\,R_R}{\psi_R^{\text{ref}}}$, as represented by the RHS bracketed term in equation (6.6c).
- `shaft_freq_n`= $\frac{p\,n_{\text{me}}}{60\,f_{\text{fs}}}$ per unit estimated electrical shaft speed of the induction machine, where p is the number of pole pairs of said machine.
- `ref_freq_n`= $\frac{p\,n_{\text{me}}}{60\,f_{\text{fs}}}$ per unit synchronous reference electrical shaft speed of the induction machine, where p is the number of pole pairs of said machine. This variable is equal to the stator frequency reference when shaft speed control is OFF.

A set of 'Rate Limiter' modules, entitled 'F-ramp' (sets the frequency ramp rate) and 'FLX-ramp' (sets the flux ramp rate), with input the per unit user

flux value `flx_n`, have been introduced in figures 6.6, 6.7 to avoid erratic changes to the user input variables (as actually used by the controller), which may damage the converter or motor. The rate of variable change is specified in the 'Operations' dialogue box (see figure 6.5).
Outputs of the 'V-f Controller' module are the modulation index variables `md_ref`,`mq_ref`, which are found by scaling the variables u_d^{ref}, u_q^{ref} by 2 times the inverse DC bus voltage. The third output present, is the per unit stator reference frequency `reffreq_n2`, which is connected to an integrator that generates the Cos/Sin angles of the reference voltage vector (see figure 6.6).

An example of drive operation is shown in figure 6.8 for two operation modes. The results given figure 6.8(a), represents operation without speed control active, i.e the shaft reference then represents the synchronous shaft speed. The quadratic load/speed characteristic is set to produce 100 mNm at 1000 rpm. Also shown in figure 6.8(b) is operation (same speed reference setting of 750 rpm), with speed control active. The sampling time T_s as defined in the program pull-down menu is set to 100 μs, i.e a 10 kHz ADC sampling frequency is used for the discrete controller model. For this example the flux reference is set to 0.03 Wb and observation of the two time plots shows that the use of a shaft speed sensor is effective in achieving accurate speed control.



(a) Shaft speed control OFF (b) Shaft speed control ON

Fig. 6.8 Simulation: Typical waveforms obtained using a V-f controller

A vector plot which corresponds to the results given in figure 6.8(b) at time ≈0.25 s, shows the controller unity d-axis 'green', current vector 'red', rotor flux vector 'brown' (scaled by a factor 100) and the terminal voltage vector 'blue'(scaled by a factor 1/3).

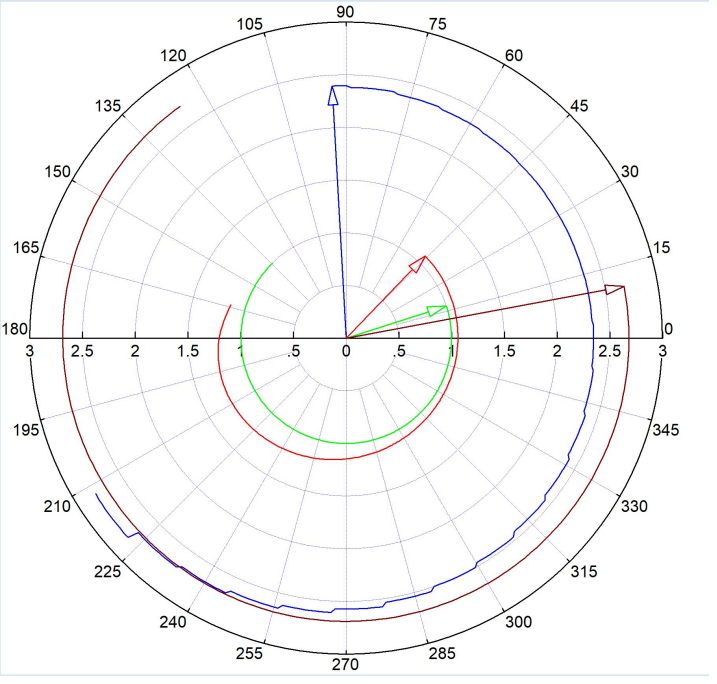


Fig. 6.9 Simulation: Vector plot showing V-f controller operation with speed control active

The results given in figure 6.9 show a slight miss-alignment between the rotor flux and the controller d-axis, which is due to the assumptions made with respect to ignoring the leakage inductance for control purposes.

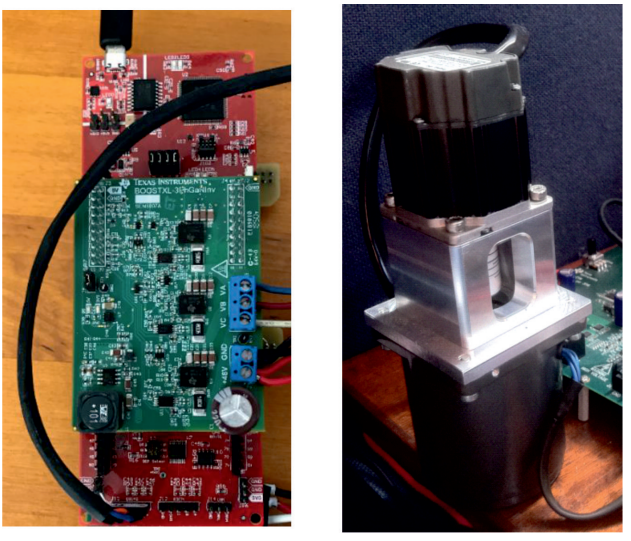
6.1.1.2 Voltage-frequency Control-F0049-GANINV:HIL

Reference files:

- `Vf_F0049-GANINV-compile.vsm`, for HIL-compile
- `Vf_F0049-GANINV-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 6.10 represents the `BOOSTXL-GANINV` converter (located on the

`LAUNCHXL-F280049` module), which is electrically connected to an induction machine [25]. A PM machine [2], is mechanically connected the induction machine, as its encoder is used for shaft speed estimation (see figure 6.10(b)).



(a) MCU (b) Motor setup

Fig. 6.10 F280049-GANINV setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. electrical wiring details are important to ensure that the direction of rotation is in accordance with controller requirements.
A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 6.11 shows the 'V-f sensored control' module, which must be compiled to generate an out-file .
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user reference speed `nref` and the reference rotor flx `Rotflx`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller (when selected). Three logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other one is used to select shaft speed control `shftspe_ctrl_on`, which activates the

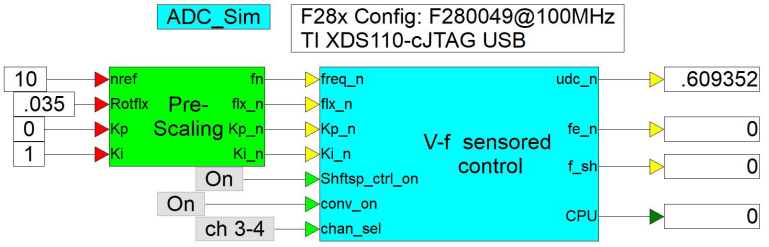


Fig. 6.11 HIL-Compile model of a Induction machine drive with Voltage-Frequency control

speed controller. When `off` the speed reference represents the 'synchronous shaft speed'. Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- A per unit synchronous speed estimate `fe_n` generated by the V-f controller.
- A per unit shaft speed estimate `f_sh` generated via the encoder.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 6.11 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 6.12.
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box, mentioned earlier (see figure 6.5), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage/frequency. The 'Converter Variables' dialogue box, also introduced earlier (see figure 4.191), defines the scaling used by the Analogue to Digital converter, as discussed earlier (see equation (4.2)).

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

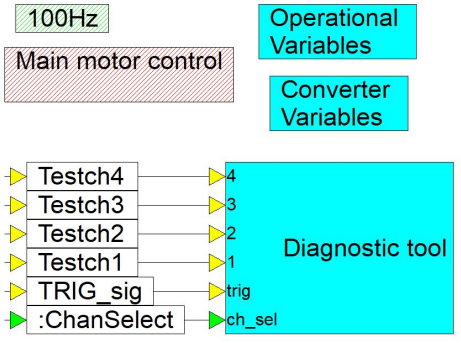


Fig. 6.12 One level down into the FOC sensored Controller module

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 6.12, is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 6.13. Of these, the 'V-f controller' has been discussed earlier (see figure 6.6), where it was used in phase, SIM of the development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages, are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F280049` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present, which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.
The `ADC-PWM-LaunchXL-28004x BOOSTXL-3phGANINV` unit is used generate the PWM signals for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are con-

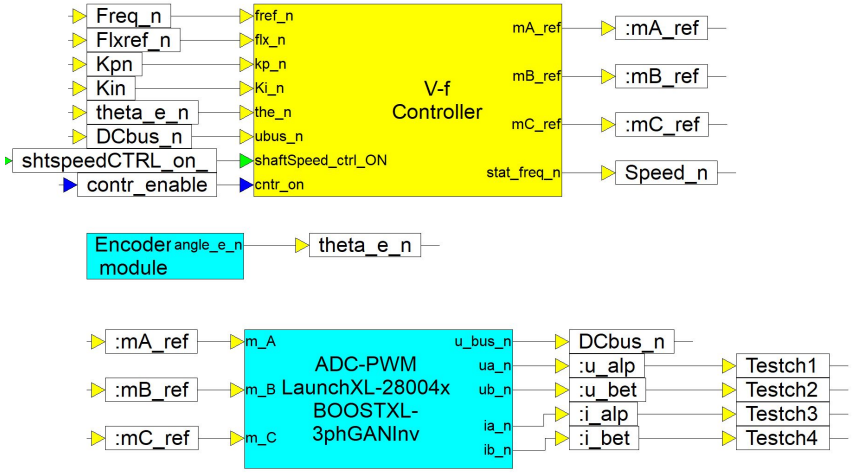


Fig. 6.13 One level down into the 'Main motor controller' module

verted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively. ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 6.14 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes said out-file. Two sliders are used to set the flux reference value and the electrical frequency. The other sliders are used for setting the proportional and integral gains of the speed controller. Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. In addition, the synchronous and measured shaft speed are converted to actual values and shown on numeric displays. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.
The plot given in figure 6.14 displays the per unit LPF measured voltage vector locus. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus. The speed reference in figure 6.14 is set to 600 rpm. These results show operation with shaft speed control active, with the machine operating under no-load conditions, as is apparent from the small difference between synchronous and measured speed.

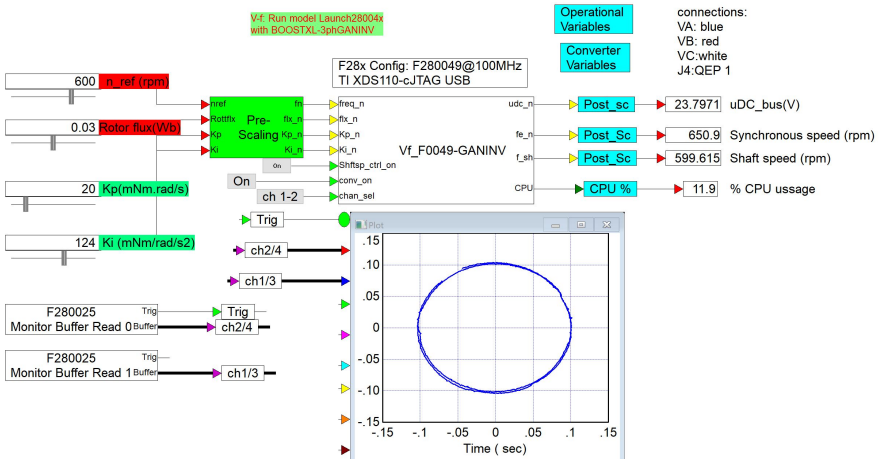


Fig. 6.14 HIL-Debug model, F0049-GANINV of an induction machine drive with shaft speed control option

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

6.2 Sensored Field Oriented Control-IM

Purpose of this section is to implement sensored rotor flux based field oriented control of an induction machine, with torque or speed control, as discussed earlier (see figure 2.23).

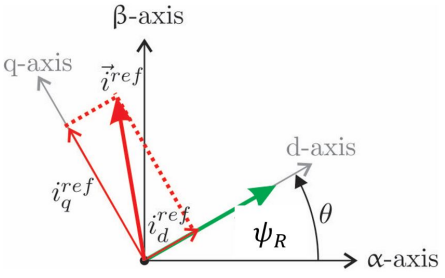


Fig. 6.15 FOC for induction machines , with reference direct axis current i_d^{ref}, quadrature current i_q^{ref} inputs, which makes use of the measured shaft angle θ_{me} to calculate the d-axis angle θ tied to the rotor flux vector

The induction machine used for this example does not have a shaft sensor, hence use is made of the encoder attached to the PM motor. Both machines must therefore be mechanically connected using the coupling assembly provided for this purpose (see figure 6.10(b)). Use is made on the current control approach discussed earlier for PM machines (see chapter 4.2). For rotor flux based field oriented control the rotor flux angle needs to be calculated, which can be realized by making use of either the measured shaft speed or shaft angle [13].

In this case the encoder shaft angle θ_{me} is used and the shaft speed ω_{me} is estimated, hence it is prudent to make use of the encoder shaft angle. However, a shaft speed sensor can also be used, in which case the slip speed must be added to the latter (after multiplication with pole pair number) to produce an estimate for the stator frequency and that must be integrated to produce the reference angle for the controller [13]. The angle estimator shown (by way of an ACTIVATE [17] model in figure 6.16), makes use of the reference direct axis current `id_ref`=i_d^{ref}, quadrature reference current `iq_ref`=i_q^{ref} and measured shaft angle `th_me`=θ_{me}.

The angle, `ref_angle`= θ^{ref}, generated by the estimator, is used by a synchronous model based current controller, as shown previously for PM machines (see section 4.2). In this case the direct axis reference current controls the rotor flux of the machine, whilst the quadrature component controls the shaft torque.

The angle estimator satisfies the following set of equations (6.7) [13].

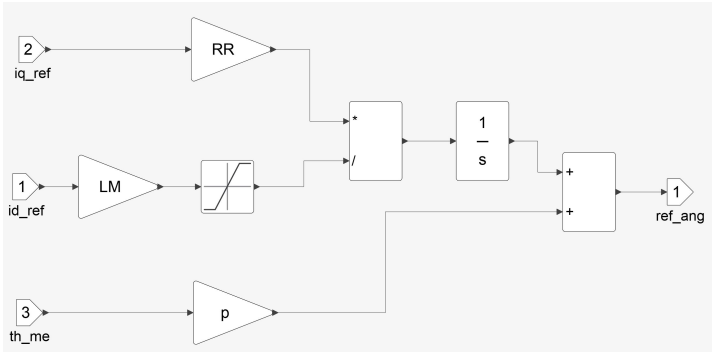


Fig. 6.16 Angle estimator for FOC induction machine control, with reference direct axis current i_d^{ref}, quadrature current i_q^{ref} inputs, which makes use of the measured shaft angle θ_{me}

$$\psi_R^{\text{ref}} = i_d^{\text{ref}} L_M \tag{6.7a}$$

$$\theta^{\text{ref}} = p\,\theta_{\text{me}} + \underbrace{\frac{i_q^{\text{ref}} R_R}{\psi_R^{\text{ref}}}}_{\omega_{\text{sl}}} \left(\frac{1}{s}\right) \tag{6.7b}$$

where L_M represents the magnetizing inductance of the four-parameter model (discussed in section 2.1.3.5), and ω_{sl} represents the slip frequency frequency variable (rad/s). This is integrated and added to the electrical shaft speed $\theta_{\text{m}} = p\,\theta_{\text{me}}$, which generates the angle `ref_ang`=θ^{ref}, which is the reference for rotor flux d-axis (in stationary coordinates). The quadrature reference current i_q^{ref}, follows from the torque expression

$$T_{\text{me}} = \frac{3}{2}\, p\, i_q^{\text{ref}}\, \psi_R^{\text{ref}} \tag{6.8}$$

where p represents the number of pole pairs of the machine and T_{me} , the mechanical shaft torque. The speed controller, with gains K_p, K_i, can in the continuous time domain be represented as

$$T_{\text{me}} = \left(K_p + \frac{K_i}{s}\right)\left(\omega_{\text{me}}^{\text{ref}} - \omega_{\text{me}}\right) \tag{6.9}$$

where ω_{me}, is the shaft speed of the machine (rad/s), that is estimated from

the shaft angle. Use of said equation in combination with equation (6.8) yields an explicit expression for the reference quadrature current i_q^{ref}, namely:

$$i_q^{\text{ref}} = \frac{2}{3\,p^2\psi_R^{\text{ref}}}\left(K_p + \frac{K_i}{s}\right)\left(\omega_{\text{me}}^{\text{ref}} - \omega_{\text{me}}\right) \tag{6.10}$$

Expression (6.10) defines the quadrature current input for the synchronous current controller based on the estimated (from an encoder angle) and reference electrical shaft speed when speed control is required. A shaft speed estimate ω_{me}, as required for speed control, is derived from the encoder angle input signal θ_{me}.

6.2.1 IM Sensored Field Oriented Control-TI

The purpose of this section is to realize sensored Field-Oriented Control of an induction machine [25] using Texas Instruments hardware. The tool chain introduced in figure 2.45 is used as a template for the ensuing subsections covering SIM→PIL→COSIM→HIL, which is based on the `F0049-GANINV` setup, but other HIL examples are also provided.

6.2.1.1 IM Sensored Field Oriented Control: F0049-GANINV:SIM

Reference program: `IM-FOC-F0049-GanINV-SIM.vsm`

This example considers the simulation of the IM motor drive in use, with a fixed point FOC and current controller as shown in figure 6.17.
The controller outputs are the three reference modulation indices `mA_ref`, `mB_ref`, `mC_ref`. These variables are connected to a 'CONV' module, which multiplies the modulation indices by the term $u_{\text{DC}}/2$. Said module also has three limiters, that limit the voltage of the motor to $\pm u_{\text{DC}}/2$. A Forward Clarke transformation is again used to generate the voltage vector for the machine model. Outputs of the machine model are the currents i_α, i_β shaft torque T_{me} shaft speed n_{me} and shaft angle θ_{me}. The latter variable is used by the controller to calculate the appropriate rotor flux angle θ^{ref} (see equation (6.7b)) for the machine. In addition, the motor model generates the rotor flux variables $\psi_{R\alpha}$, $\psi_{R\beta}$, which are used to show the scaled rotor vector in a

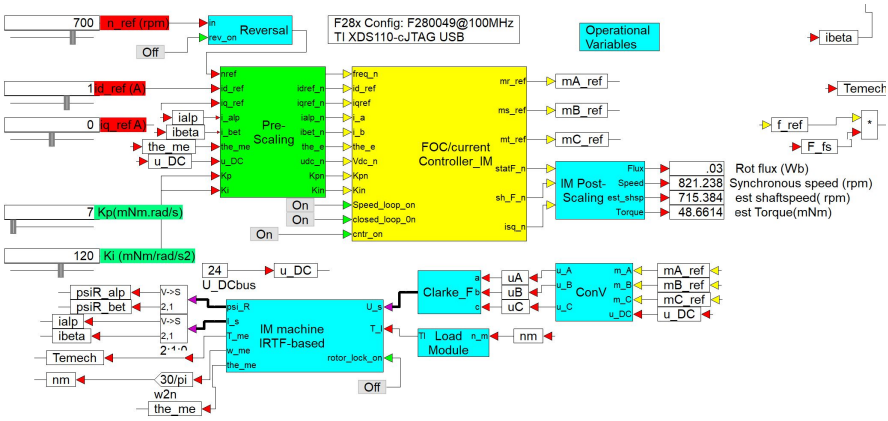


Fig. 6.17 Simulation of an induction drive with a FOC/Current Controller

polar plot. A set of sliders is used to define the current reference vector, in terms of its direct and quadrature components i_d^{ref}, i_q^{ref} and rotational speed $n_{\text{me}}^{\text{ref}}$. A 'pre-scaling' module has been added to convert the floating point variables required for the controller to fixed point format. In addition, two sliders are provided, which allow the user to set the proportional K_p^{sp} and integral K_i^{sp} speed gains of the controller. Note that said gains are shown in (milli)Nms/rad and (milli)Nm/rad respectively. A guideline for choosing these parameters is given in section 2.1.2.
The dialogue box 'Operational Variables' for this example (see figure 6.18), contain all the user settings required. This dialogue box is again used to assign

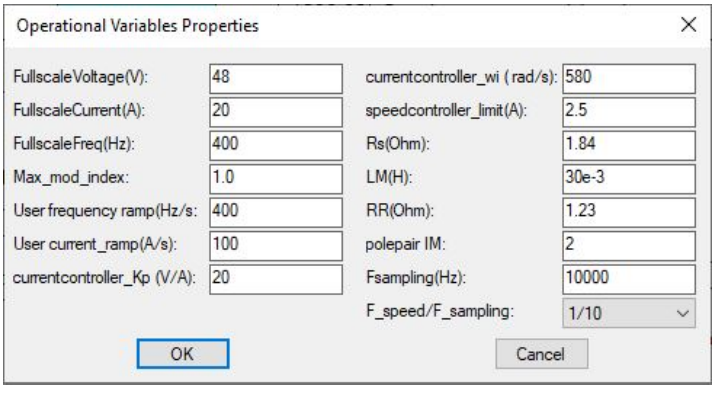


Fig. 6.18 Simulation: Dialogue box 'Operational Variables'

full scale values for the voltage, current and frequency. In addition, the current controller gain and bandwidth are also assigned here. These values are based

on the motor in use, as was discussed in section 2.1.4. A dialog box entry for the maximum modulation index value and speed controller limit value `speedcontroller_limit(A)` are also shown. The latter value is the maximum (absolute) quadrature current value, which the speed controller is assigned to provide. Note that the maximum modulation index value corresponds to a circular limit in the α, β coordinate system, of which the radius is set by said variable.
Also assigned in this box are the rate of variable change for current and frequency, i.e. introduced as a safety measure to avoid erratic slider action from causing over-currents in the drive. In addition, this dialogue box assigns the motor pole pair number and required motor parameters (four parameter model representation (see figure 2.25)) of the induction machine in use. Furthermore, the sample frequency used for the speed controller module is set in terms of a ratio relative to the controller sample frequency, which is also indicated in this dialogue box for convenience. Note that the actual controller sampling rate is set in the Embed 'Systems property dialogue box'.

Moving one level lower into the 'FOC/Current Controller IM' module reveals a set of modules given in figure 6.19.

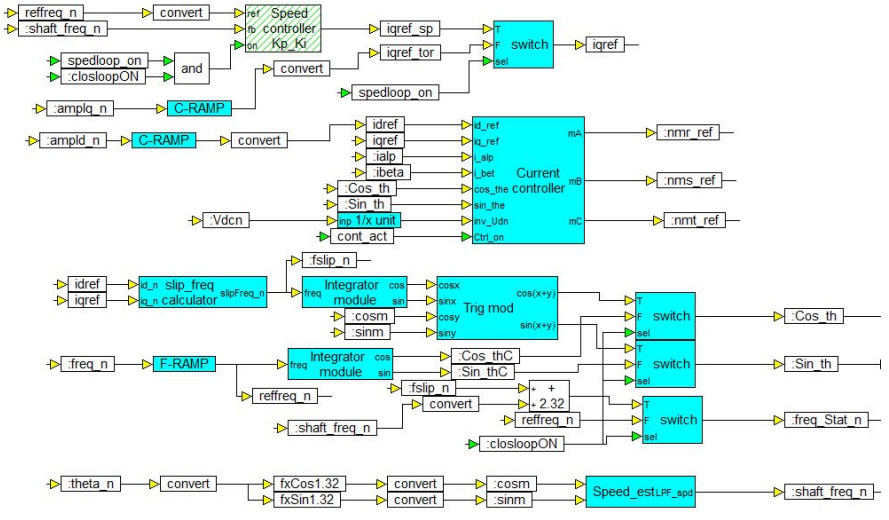


Fig. 6.19 Fixed point representation of the IM FOC/Current and speed controller

Synchronous model based current control is undertaken by a 'Current Controller' module, which has as inputs the per unit quadrature and direct axis reference value, measured α, β currents, Cos/Sin inputs of synchronous reference frame and inverse DC bus voltage. The latter is generated by the '1/x ' module, which generates the per unit inverse DC bus voltage using a numerical routine, as discussed in previous examples. A 'switch' module,

activated by the `closloopON` signal (active if the shaft encoder is required) is used to select which Cos/Sine variables are to be used by the current controller module. When this switch is `OFF`, operation of the drive is under open loop current & frequency control, which is a good initial step to verify operation of the current controller. When `closloopON` is ON a 'slip freq calculator' and 'integrator' input are used in combination with a `Trig module` to generate the Cos/Sin signals for the current controller. The function of said module'is to generate the variables $\cos(\theta_{\text{slp}} + \theta_m)$, $\sin(\theta_{\text{slp}} + \theta_m)$. Inputs to this module are the cos/sine of the slip frequency and the Cos/Sine of the electrical shaft angle: `cosm`, `sinm`.

Inputs to the speed controller are the (rate limited) reference frequency `reffreq_n` and
estimated shaft frequency (from the 'speed est' unit) `shaft_freq_n`. Output of the speed controller module (when enabled) is the per unit quadrature reference current `iqref_sp`. Note that the speed controller module is shown 'hatched green', which implies that its sample rate is different to the remaining part of the controller. In this case the sample frequency for the speed controller is set to 1 kHz. For less time demanding activities it is prudent to use a lower sampling frequency, which serves to avoid MCU overrun errors that can occur when insufficient time is available to execute all the required function for a given ADC frequency. A 'switch' module with output `iqref` determines whether the quadrature reference current for the current controller is provided by the speed controller or the user i_q^{ref} slider. The switch in question is controlled by the `speedloop_on` signal (active when closed speed-loop control is required, 'off' implies operation in 'torque mode').

A set of 'Rate Limiter' modules, entitled 'C-ramp' (sets the current ramp rate) and 'F-ramp' (sets the frequency ramp rate), have been introduced in figure 6.19 to avoid erratic changes to the user input variables (as actually used by the controller), which may damage the converter or motor. The rate of variable change is specified in the 'Operational Variables' dialogue box (see figure 6.18).

An example of drive operation is shown in figure 6.20. The sampling time T_s as defined in the program pull-down menu is set to 0.1 ms, i.e a 10 kHz sampling frequency is used for the discrete controller model. A quadratic load torque/speed characteristic is used which provides a torque of 100 mNm at 1000 rpm.
The top plot of figure 6.20, shows the motor phase current and phase voltage using sensored, field-oriented, speed control, which is initiated at $t \approx 0.1$ s. The delay in speed control operation, was purposely done to ensure that the induction machine flux level is at its rated level 30 mWb, before speed control is undertaken. In this case the simulation was run over a 0.3 s time interval, with a motor inertia J=$86.6 \cdot 10^{-6}$ kgm^2 (taken to be the estimated combined inertia of the PM/IM motor unit). The lower sub-plot shows the actual and

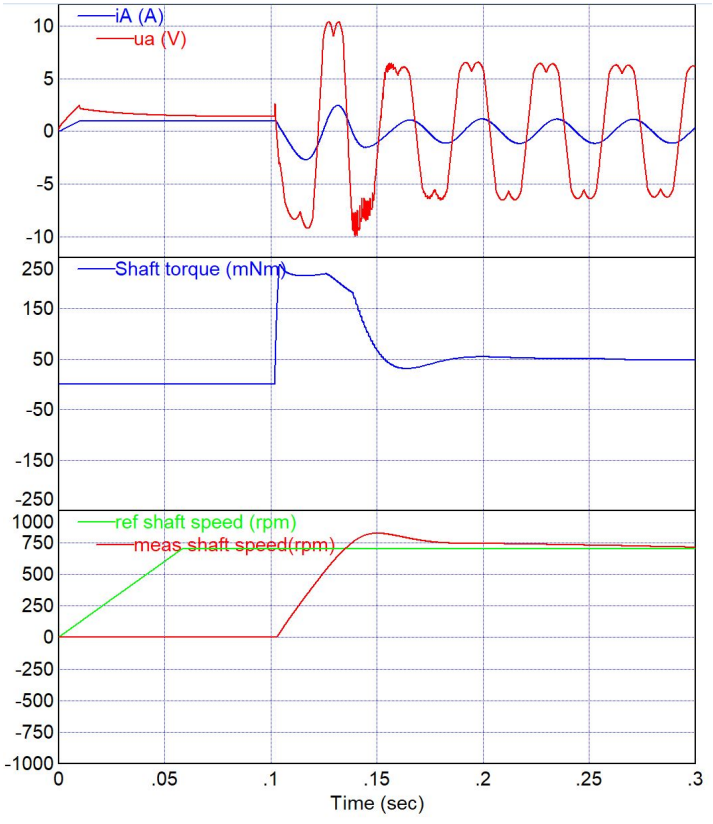


Fig. 6.20 Simulation: Typical waveforms obtained using a FOC/speed IM controller

reference shaft speed for a given frequency ramp setting (as set in the dialogue box). Also shown in this figure is actual shaft torque of the machine over the 0.3 s simulation interval.
Some interesting observations can be made from the results shown in figure 6.20 during operation namely:

- the maximum phase current is determined by the maximum $i_q^{\max}$ value set in the speed controller and the chosen direct axis reference current value. Hence with $i_d^{\text{ref}} = 1.0$, $i_q^{\max} = 2.5$ the maximum peak phase current will be equal to $i_s^{\max} = \sqrt{1.0^2 + 2.5^2}$=2.69 A.
- The phase current will be determined by the selected direct axis reference current value and quadrature current required for the load, when the shaft speed is constant.
- The maximum shaft torque during startup can be written as $1.5\, p\, L_M\, i_d^{\text{ref}}\, i_q^{\max}$, which is equal to 225 mNm in this case, given the chosen direct axis reference current setting of 1.0 A.

Results shown above relate to operation under speed control, but this example can also be operated in two alternative modes, which can be set by the user namely:

- `Speed_loop_on` button 'off': speed loop disabled, in which case torque control is activated, i.e the i_q^{ref} slider set the torque. Note that a 'speed runway situation' (shaft speed increases to high value) can occur, as torque is (in theory) independent of the shaft speed.
- `Closed_loop_on` button 'off': operation without a shaft speed sensor, hence speed is dictated by the shaft speed reference slider, which then sets the synchronous shaft speed. Operation is then under current& frequency control, as mentioned earlier.

In the sequel to this section a vector plot is shown of the drive in question operating under steady state conditions, with a constant speed of n^{ref}=700 rpm and arbitrarily chosen mechanical load of 48.6 mNm at time mark ≈0.3 s.

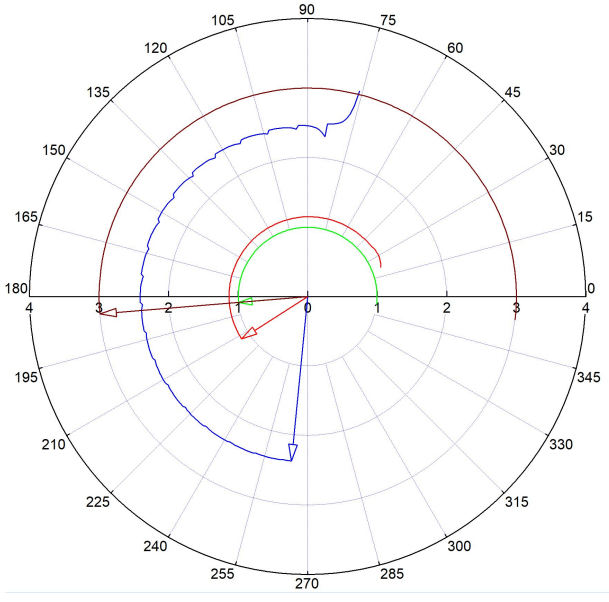


Fig. 6.21 Simulation: Vector plot obtained using a FOC/Speed controller with current $\vec{i}_s$ ('red'), voltage vector $\vec{u}_s/3$ ('blue), rotor flux vector $\vec{\psi}_R \times 100$ ('brown') and unity d-axis vector ('green')

Said vector plot (see figure 6.21), shows the current vector $\vec{i}_s$ and voltage vector $\vec{u}_s$, whereby the latter is scaled by a factor 3. In addition, a unity 'd-axis' vector is introduced to show the orientation of the d,q reference frame, where the positive d-axis is displayed as a 'green' vector, which is aligned with

the rotor flux vector. Its magnitude is set by the product of the direct axis current and magnetizing inductance L_M. In this example, a current i_q value is generated given that the machine is producing torque to match the load torque applied. Consequently the current vector orientation relative to the d-axis vector is defined by the direct axis current value i_d and the quadrature current i_q, as is indeed the case. The voltage vector is made up of a back EMF component that orthogonal to the d-axis vector and resistive voltage component, which is aligned with said current vector. The third component of the voltage vector is due to the leakage inductive component of the machine which is orthogonal to the current vector and its value will depend (among others) on the electrical frequency. Finally the (scaled) rotor flux vector generated by the IRTF motor model is also added and this is aligned with the controller 'd-axis' (as anticipated because the direct axis current generates the rotor flux).

6.2.1.2 IM Sensored FOC with Enc-F0049-GANINV: PIL

Reference files:

- `IM-FOC-F0049-GanINV-PIL.vsm`
- `IM-FOC-PIL.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, this entails that the controller module shown in the simulation (see figure 6.17), is now replaced with a target execution module for file: `IM FOC_PIL_mod.out`, as indicated in figure 6.22.

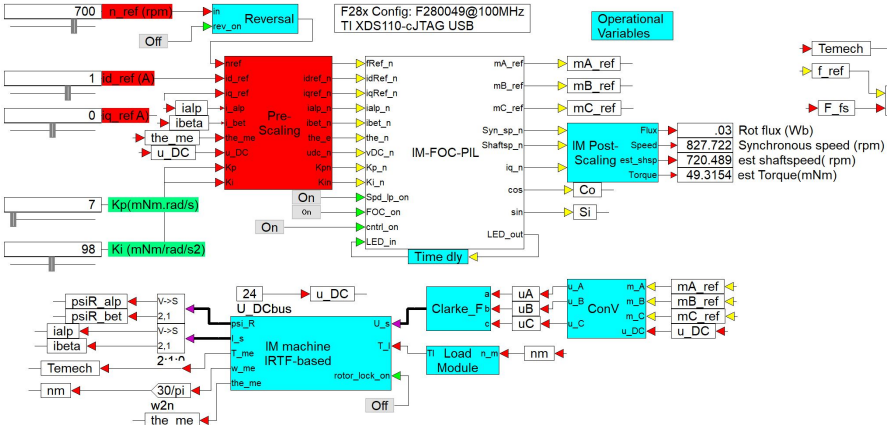


Fig. 6.22 IM Field-oriented torque/speed control with encoder: PIL operation

The target interface model that executes the out-file, generates the modulation indices required for the converter motor and these components are not affected by use of the PIL. Generation of the out file is based on the PIL module (see figure 6.23), which must be compiled using the command, `toolbox>codegen`.

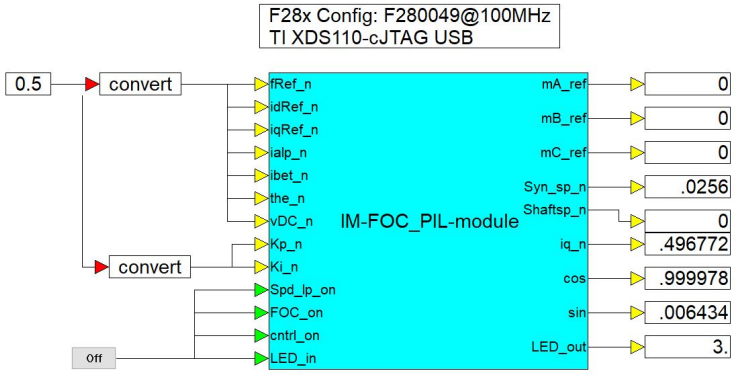


Fig. 6.23 PIL module, IM FOC control

Moving one level into the PIL module `IM-FOC_PIL_mod` shows the FOC/speed controller as introduced in section 6.2.1.1.

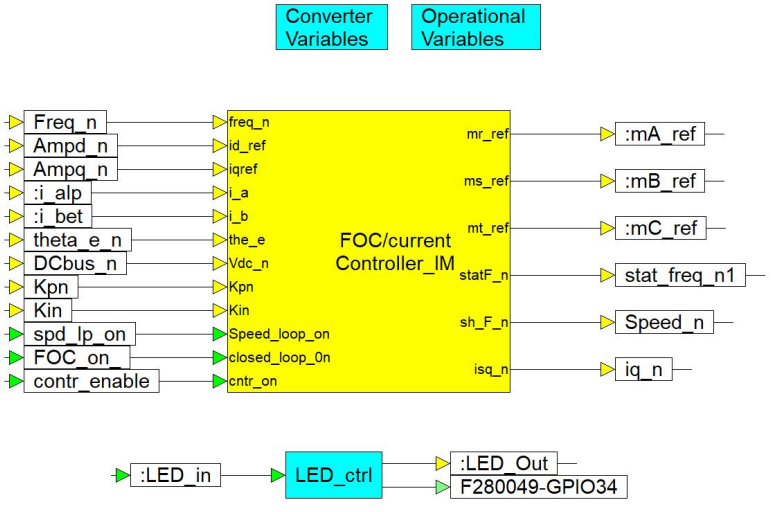


Fig. 6.24 Content PIL module, IM Field-oriented torque/current and speed control with encoder

Added to the FOC controller is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that the PIL control is operational. In addition, two variables `cos`, `sin` are provided as outputs and these are used in a vector plot to show the 'd' axis of the controller algorithm.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational. Results produced by PIL operation (running in this case with FOC-speed control active) are identical to the plots shown in figures 6.20, 6.21 and are therefore not shown here.

6.2.1.3 IM Sensored FOC with Enc-F0049-GANINV: COSIM

Reference files:

- `IM-FOC-enc_F0049_GANINV-COSIM.vsm`
- `GAN_INVv13.psimsch`
- `IM motor model_v1.psimsch`

To evaluate operation of a controller with a switching converter, in this case the `GANINV`, use can be made of a `PSIMCoupler` module, as indicated in figure 6.25. This approach implies the use of PSIM together with Embed, which is referred to as 'co-simulation'.

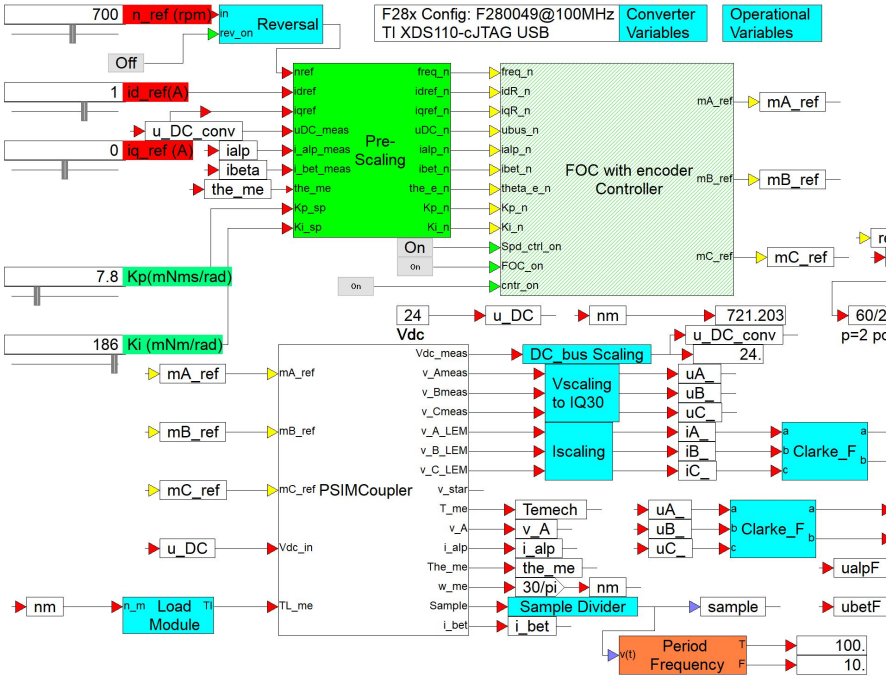


Fig. 6.25 Cosim based IM drive, field-oriented torque/current and speed control with encoder

The `FOC with encoder controller`, hatched in this case, as its sampling frequency is locally set to 10 kHz, whereas the 'system' frequency (set in the 'system properties' dialogue box) is set to 20 MHz. The latter is much higher, given the need to accurately represent 20 kHz voltage switching waveforms generated by the PWM unit of the converter. The content of

the `FOC with encoder controller` module was introduced earlier (see figure 6.19). However, the current and voltage measurements now make use of the signal processing circuitry connected to the half-bridges.
In this case the current transducers are in the converter phases, as shown in figure 6.26. After sampling and signal-processing in the converter, these variables appear as: `iA_`, `iB_`, `iC_` in figure 6.25. The low pass filtered voltages (which removes the PWM component) of the three half-bridges, is also present in figure 6.25, and are scaled and sampled by the module `Vscaling to IQ30`. In this case the per unit variables are rescaled to actual values: `uA_`, `uB_` and `uC_` respectively.
The 'Converter Variables' dialogue box (see figure 6.25), defines the scaling used by the Analog to Digital converter, which has a maximum input value of 3.3 V and is directly linked to the signal processing undertaken for voltages and currents as discussed above. For example, the 'ADC-fullscale-voltage' value is the theoretical peak-to peak input voltage which must be applied to the low-pass filter network (see figure 4.12) to ensure that the ADC input level varies between $0 \rightarrow 3.3$V.
Apart from the measured voltage/currents generated by the converter, the PSIM coupler unit also provides the actual motor model current vector: `i_alp`, `i_bet` (in stationary coordinates), phase 'A' voltage and shaft speed, which are presented by the variables `v_A` and `w_me` respectively (see figure 6.26).

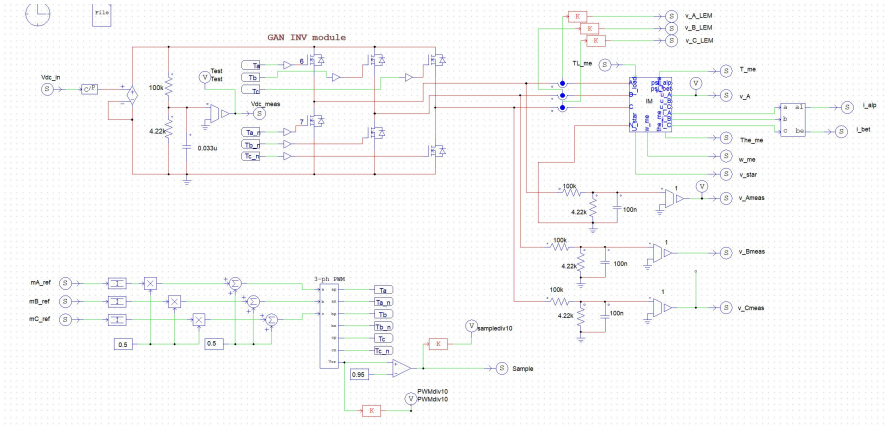


Fig. 6.26 Content PSIMcosim model of GANINV and induction machine

The PSIM induction machine model `IM` used in the coupler module (see figure 6.27), is based on the IRTF model discussed earlier (see figure 2.25).

Results achieved with the COSIM approach are shown in figure 6.28.
Indicated in said figure is the phase 'A' voltage waveform together with the

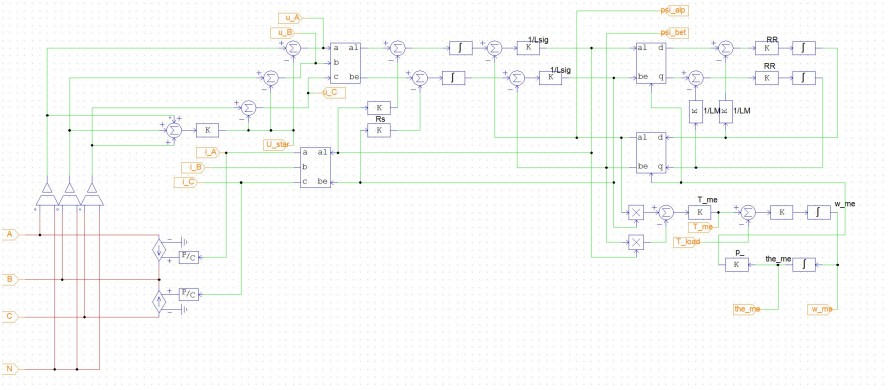


Fig. 6.27 PSIM model, four parameter squirrel-cage induction machine

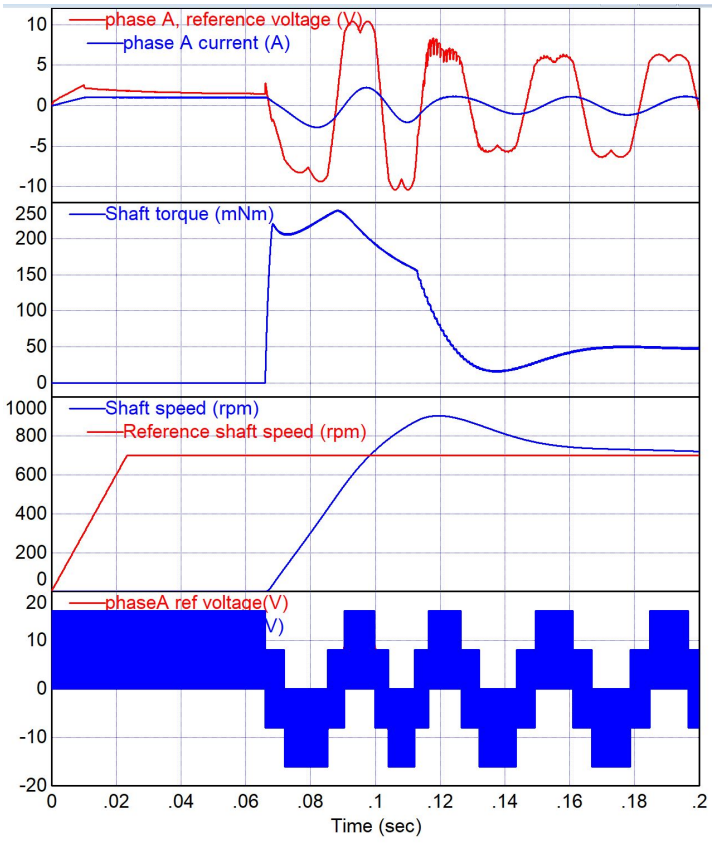


Fig. 6.28 Result COSIM, IM field-oriented torque/speed control with encoder

reference voltage, phase current, shaft torque and shaft speed. Note that speed control is activated around time mark 0.065 s, to build the machine rotor flux to an acceptable level first.
A vector plot (see figure 6.29), at time mark 0.2 s after starting the co-simulation, shows the measured LPF voltage vector ('blue' and scaled by a factor 1/5), unity d-axis vector ('green') and the actual (from the IRTF motor model) current vector ('red').

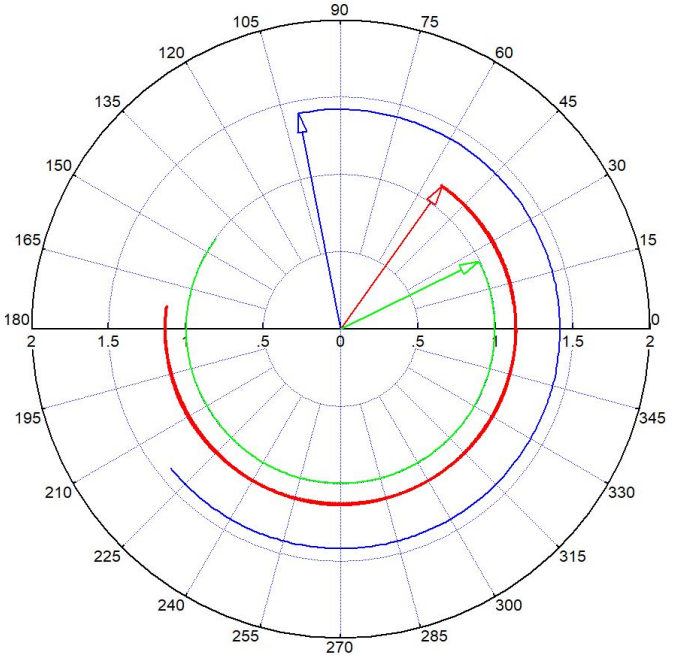


Fig. 6.29 Result COSIM, IM field-oriented torque/speed control with encoder: vector plot

Co-simulation, as discussed in this sub-section, is instructive for understanding the impact of using a switching converter on drive operation, e.g. to assess the impact of shaft torque ripple due to PWM.

6.2.1.4 IM Sensored FOC with Enc-F0049-GANINV: HIL

Reference files:

- `IMFOC_sensored_F0049-GANINV-compile.vsm`, for HIL-compile
- `IMFOC_sensored_F0049-GANINV-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), shown in figure 6.30, represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F280049` module), which is connected to an induction machine[25]. In addition, this motor is also mechanically connected a PM machine[2] because use is made of its encoder for sensored IM FOC operation.

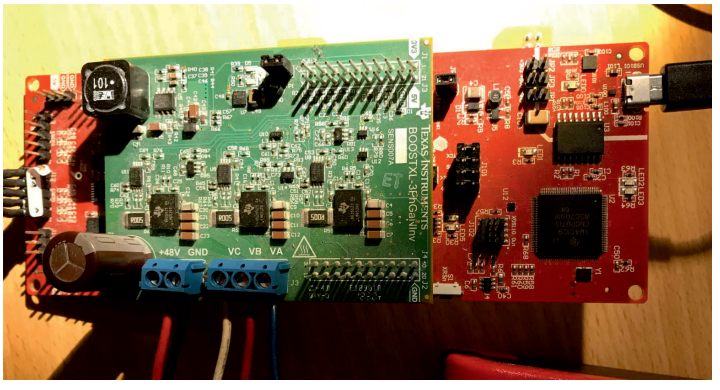


Fig. 6.30 F280049C-GANINV setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs, which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration.

The model given in figure 6.31, shows the 'IM FOC sensored and C&F control' module, which must be compiled to generate the file `XXX.out`.
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct/quadrature axis reference currents `id`, `iq` and the reference shaft frequency `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller.
Four logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be

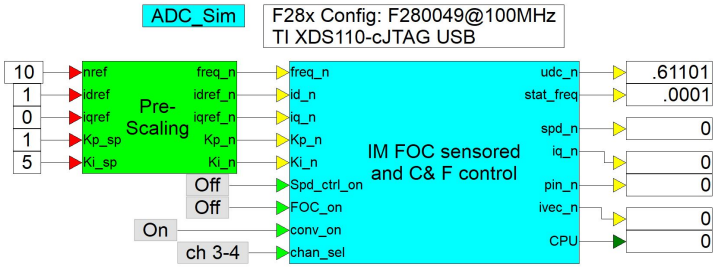


Fig. 6.31 HIL-Compile model of an induction machine drive with field-oriented control and encoder

viewed. The other two are used to select field-oriented control (FOC) operation `FOC_on`, and speed control operation using FOC `Spd_ctrl_on`, as discussed earlier (see section 6.2.1.1).
Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- A per unit synchronous speed estimate `stat_freq` generated via the controller.
- A per unit shaft speed estimate `sp_n` generated via the encoder.
- The measured per unit quadrature current `iq_n`, used to generate a shaft torque estimate.
- A per unit estimate of the motor input power `pin_n`. This is also useful as an estimate for the drive input power (but this ignored the converter losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx p_{\mathrm{in}}/u_{\mathrm{DC}}$.
- The measured per unit current vector amplitude `ivec_n`.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 6.31 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Hence these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes shown in figure 6.32.
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation.
The 'Operational Variables' dialogue box (see figure 6.33), is used to assign full

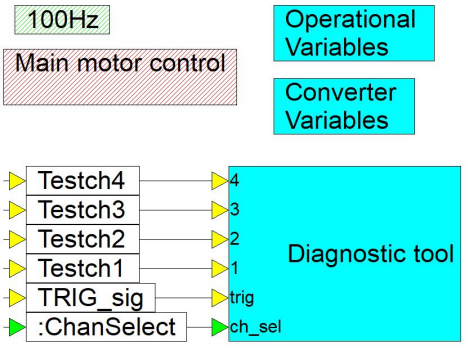


Fig. 6.32 One level down into the IM FOC sensored Controller module

scale values for the voltage, current, frequency and the ramp change for the reference current(s)/frequency. In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the speed controller sampling frequency is typically set lower then used by the main controller. In this case it is set to 1 kHz. Also added to this dialogue module are motor parameters: motor pole pair number p, magnetizing inductance L_M and rotor resistance R_R of a four parameter induction machine model.
Details on how these parameters can be found will be discussed later (see 'Post Analysis', section 6.2.1.4).

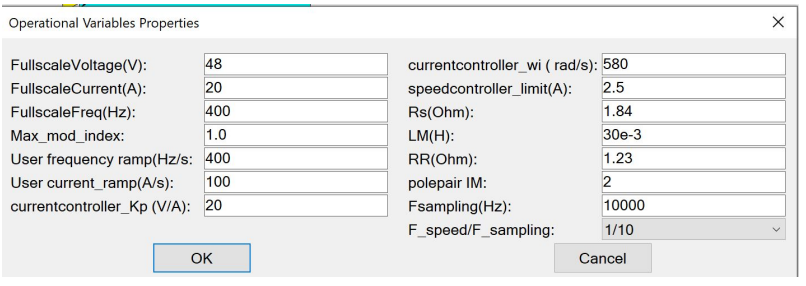


Fig. 6.33 HIL: Dialogue box 'Operational Variables'

The 'Converter Variables' dialogue box (see figure 6.34), defines the scaling used by the Analogue to Digital converter, as discussed earlier (see equation (4.2)).
This dialogue box also shows the low-pass filter corner frequency used by the converter unit for measuring/filtering the half-bridge voltages (see figure 4.12).

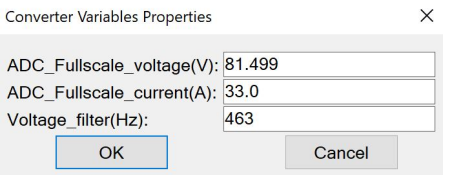


Fig. 6.34 HIL: Dialogue box 'Converter Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 6.32 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit.
Moving one level lower into the 'Main motor control' module reveals a set of modules, shown in figure 6.35. Of these, the 'FOC/speed controller IM' has been discussed earlier (see figure 6.19), where it was used in phases : SIM, PIL and COSIM of the development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module.
This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F280049` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present, which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.
The `ADC-PWM-LaunchXL-28004x BOOSTXL-3phGANINV` unit is used generate

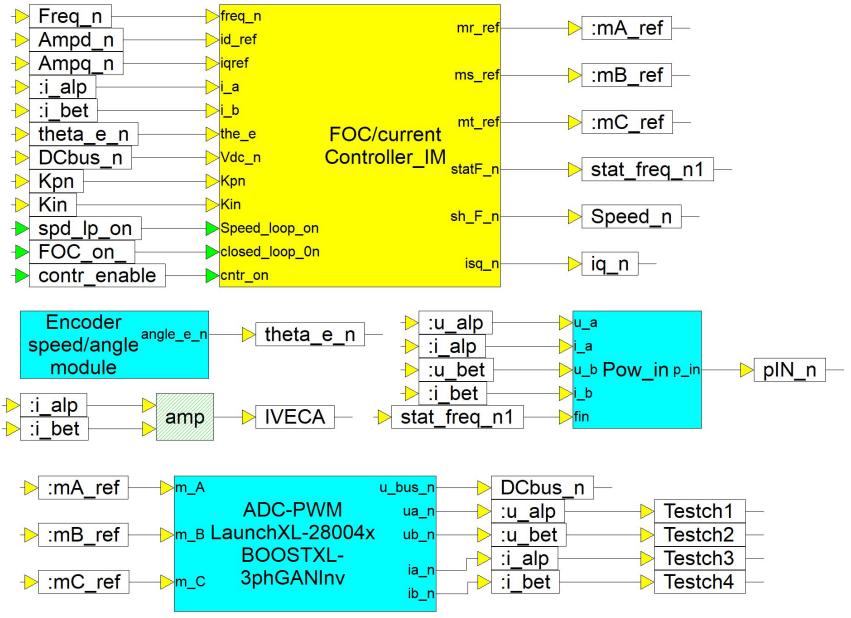


Fig. 6.35 One level down into the 'Main motor control' module

the PWM signals for the converter. In addition it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively. ADC offsets in this unit are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example (see figure 6.36) and its task is to execute the `XXX.out` file compiled in the 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct/quadrature axis current reference values and the shaft speed reference. In addition, two sliders are present, which allow the user to set the proportional and integral gain of the speed controller.
Post-scaling modules are used to convert the per-unit measured DC bus voltage, estimated rotor flux, synchronous shaft speed, measured shaft speed, shaft torque, input power and current vector amplitude to actual values, as

shown with numeric displays. In addition, the `%MCU active` display shows the % MCU activity, relative to the sampling time of the converter.
Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `chan_sel` input) diagnostic signals.

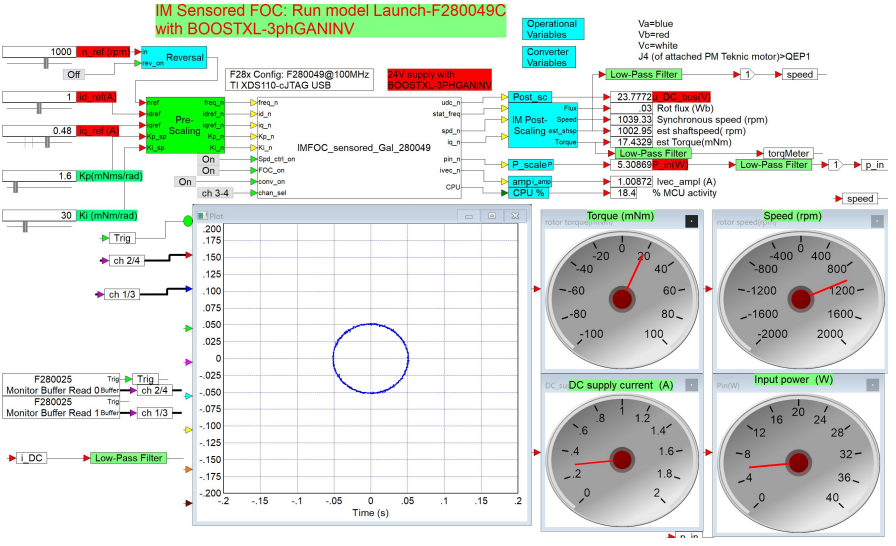


Fig. 6.36 HIL-Debug model, F0049-GANINV of a induction machine drive with field-oriented control using an encoder

The plot given in figure 6.36 displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit LPF voltage vector locus. The electrical shaft speed reference in figure 6.36 is set to 1000 rpm.
These results shows operation under FOC-speed control with the machine operating under no-load conditions. An instructive exercise for the user is to switch operation to FOC 'torque control' (implies setting `SP-ctrl-on` input to OFF) with the direct axis current set to produce rated rotor flux 30 mWb and quadrature current reference initially set to zero. Then hold the rotor by 'hand' and feel the effect of torque application when moving the quadrature reference slider.

Post analysis using Current and freq control

The application of field-oriented control for induction machine drives requires knowledge of several critical parameters of the machine and this issue is raised in the 'post-analysis' section here. Firstly, the gain and bandwidth of the synchronous current controller, need to be known, which requires access to inductance and resistance values, as discussed earlier (see section 2.1.4).

For an inductance machine, the inductance represents the leakage inductance of the (four pole) model. This can be measured by using an L-C-R meter (set to 100 Hz) and measuring across two terminals and then halving the result. Likewise the same L-C-R meter can be used to measure the stator resistance, again by measuring across two phases and halving the result. Alternatively, a DC current can be injected and Ohm's law may be applied by measuring the voltage across two phases (and then halving the result). For induction machines the total resistance assumed for current control is equal to $R_s + R_R$ and the rotor resistance needs to be defined. For current control applications here, it is sufficent to simply assume in the first instance that $R_R \approx R_s$. These two parameters: resistance $2R_s$, L_σ, together with the sampling frequency define the gain and bandwidth of the current controller.

The next step is to operate the drive under open-loop current and frequency control with the gain and bandwidth settings introduced above. The first stage is to consider no-load operation, which corresponds with the vector plot shown in figure 6.37.

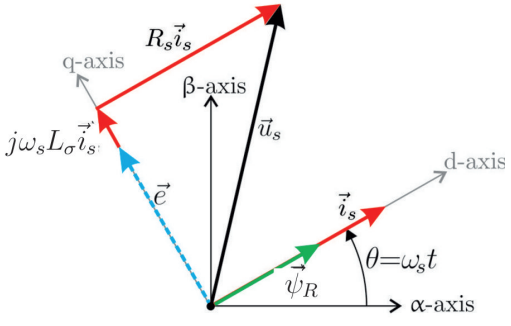


Fig. 6.37 Vector plot of an induction machine drive using C&F control: no-load operation

The important observation from this diagram is that the rotor flux and current vectors will be aligned (because no torque is produced) in which case 'Pythagorus theorum' can be applied to determine the EMF amplitude based on measurement of the LPF voltage vector amplitude, which can be found by observing the voltage locus in the HIL (with C&F active and selecting `Ch 1-2`). The EMF is then found using:

$$u_s^2 = (e + i_s\,\omega_s\,L_\sigma)^2 + (i_s\,R_s)^2 \tag{6.11}$$

For the drive operating under no-load at 1000 rpm with C&F active the following is found:

- No-load current vector amplitude $i_s = 1.0$A, measured terminal voltage $u_s = 7.91$V, electrical stator frequency $f_s = 33.3$Hz, input power $p_{in}^{\text{noload}} = 4.7$W.

- Measured (with L-C-R meter) stator resistance $R_s = 1.84\Omega$, leakage inductance $L_\sigma = 6.22$mH.

- The EMF is then equal to $e = 6.45$V, which corresponds to a rotor flux $\psi_R = {}^{e}/_{(2\pi f_s)} = 30.8$ mWb. This corresponds to a magnetizing inductance of $L_M = {}^{\psi_R}/_{i_s} = 30.8$mH.
- The iron losses under no-load are equal to $p_{fe} = p_{in} - \frac{3}{2}\, i_s^2\, R_s = 1.94$W.

Estimating the rotor resistance can be done by maintaining the same current/shaft speed settings AND locking the rotor shaft. Under those conditions the following observations can be made:

- Short circuit voltage $u_s^{\text{sc}} = 3.36$V, input power $p_{in}^{\text{sc}} = 5.0$W.
- the iron losses of the machine under locked rotor condition can be found using $p_{fe}^{\text{sc}} = p_{fe}^{\text{noload}} \left(\frac{u_s^{\text{sc}}}{u_s^{\text{noload}}} \right)^2 = 0.344$ W.
- The total copper losses are equal to $\frac{3}{2}\,(i_s)^2\,(R_s + R_R) = \left(p_{in}^{\text{sc}} - p_{fe}^{\text{sc}} \right)$. Therefore the total resistance is equal to $R_s + R_R = 3.11\Omega$, hence $R_R = 1.27\Omega$.

Note that when considering sensorless FOC, a more convenient approach is applicable to find said rotor resistance (as is discussed in section 6.4).

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

6.2.1.5 IM Sensored FOC with Enc-F069-HV kit: HIL

Reference files:

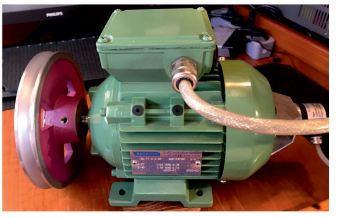
- `IMFOC_sensored_F069-HVkit-compile.vsm`, for HIL-compile
- `IMFOC_sensored_F069-HVkit-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 6.38 represents the `Texas Instruments HV kit` with `LAUNCHXL-F28069` ISO card). The converter of the `HV kit` is connected to a 0.37 kW three-phase squirrel cage induction machine [6] provided by 'Zener Electric', Sydney, Australia. Note that the motor is provided with an incremental shaft encoder, which allows sensored FOC operation, as will be discussed in this subsection. The converter of this Texas Instruments 'High Voltage Control and PFC Developers kit' [22] is connected to a 350 V, 2 A DC power supply. On the basis of name plate data [6] the following information can be deduced:

- Star connected, 2 pole pair machine.
- Peak phase voltage/current: 339 V/1.4 A.
- Rated shaft torque/speed : 2.65 Nm/1330 rpm.
- Rated stator frequency: 50 Hz.

Measurement using an L-C-R meter connected to two terminals and then halving the results yields an estimate for the phase resistance R_s=7.3 Ω and leakage inductance L_σ=75 mH. These are required together with the controller sampling frequency, for calculation of the current controller gain and bandwidth (see subsection 2.1.4).
For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 6.39 shows the 'IM FOC sensored and C&F control' module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes) for this

(a) Texas Instruments HV kit



(b) ROTOTECH Three-phase 0.37 kW IM

Fig. 6.38 MCU-converter setup for HIL operation

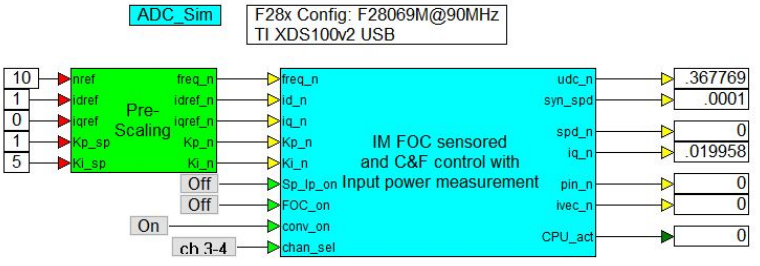


Fig. 6.39 HIL-Compile model of a induction machine drive with field-oriented control and encoder

controller module are the user direct/quadrature axis reference currents `id_n`, `iq_n` and the reference shaft frequency `freq_n`, which have been converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_n`, `Ki_n` are provided to tune the speed controller. Four logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other two are used to select field-oriented control (FOC) operation `FOC_on` and speed operation (using logic variable `Spd_lp_on`), as discussed earlier (see section 6.2.1.1).
Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- A per unit synchronous speed estimate `syn_spd` generated via the controller.
- A per unit shaft speed estimate `sp_n` generated via the encoder.
- The measured per unit quadrature current `iq_n`, used to generate a shaft torque estimate.
- A per unit estimate of the motor input power `pin_n`. This is also useful as an estimate for the drive input power (bus this ignored the converter

losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx p_{\mathrm{in}}/u_{\mathrm{DC}}$.
- The measured per unit current vector amplitude `ivec_n`.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 6.39 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes (see figure 6.40).

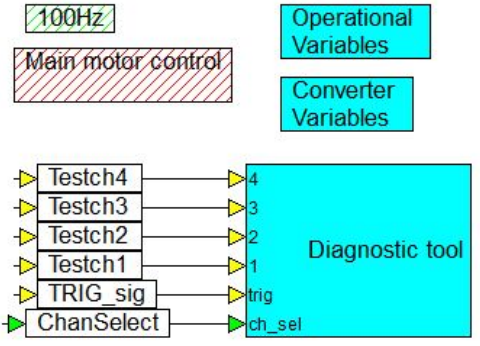


Fig. 6.40 One level down into the IM FOC sensored Controller module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 6.41), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current(s)/frequency. In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the speed controller sampling frequency is typically set lower then used by the main controller. In this case it is set to 1 kHz. Also added to this dialogue module are motor parameters: motor pole pair number p, magnetizing inductance L_M and rotor resistance R_R of a four parameter induction machine model. Identification of said parameters can in fact be done more effectively by running the drive under sensorless FOC control (see section 6.4.1.5) and this also allows identification of the magnetization characteristic of the machine (as required to determine a value for L_M, see figure 6.93). Note that FOC sensored control makes use of an approach that calculates the stator (synchronous) frequency,

based on the use of the measured shaft speed. Sensorless FOC control is the other way around, as is makes used of FAST®or AFT®, which are algorithms that make use of the stator (synchronous) frequency. Consequently, a shaft speed estimate is derived from said variable by identifying the slip frequency component.

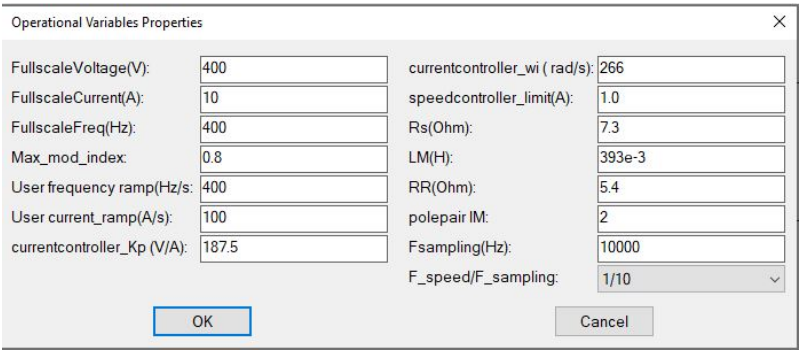


Fig. 6.41 HIL: Dialogue box 'Operational Variables'

The 'Converter Variables' dialogue box (see figure 6.42), defines the scaling used by the Analogue to Digital converter.
This module also shows the low-pass filter corner frequency used by the converter for measuring the half-bridge voltages.

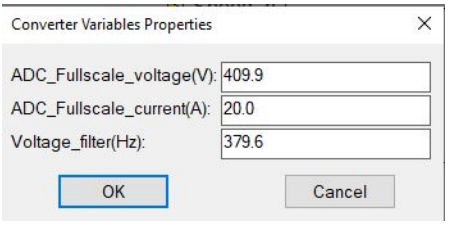


Fig. 6.42 HIL: Dialogue box 'Converter Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC

offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 6.40 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor control' module reveals a set of modules, shown in figure 6.43. Of these, the 'FOC/current controller IM' has been discussed earlier (see figure 6.19) where it was used in phases : SIM, PIL and COSIM of the development tool-chain (for a different converter setup).
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 10 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 2 μs
- Board in use: `F28069` ISO control card with `TI HV kit` based converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

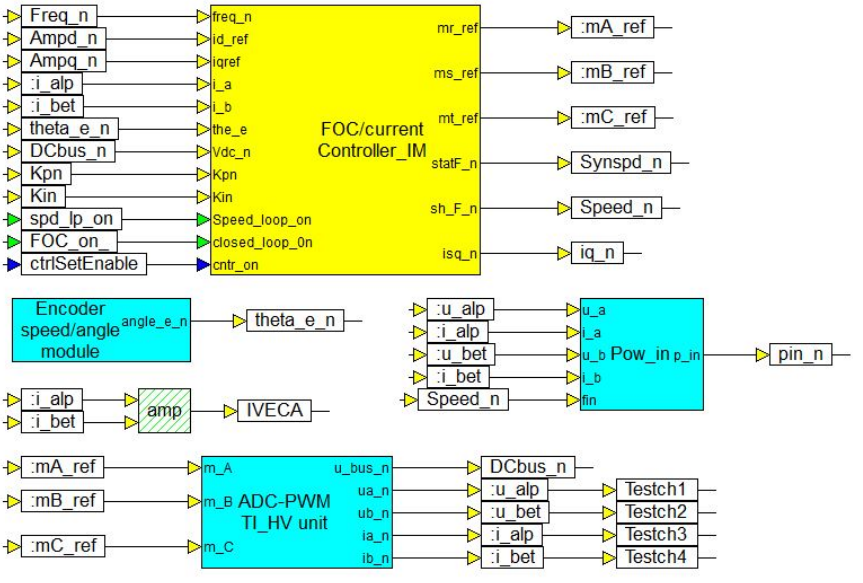


Fig. 6.43 One level down into the 'Main motor controller' module

The `ADC-PWM-TI_HV unit` unit is used generate the PWM signals for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example (see figure 6.44) and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. Three sliders are used to set the direct/quadrature axis current reference values and the shaft speed. Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

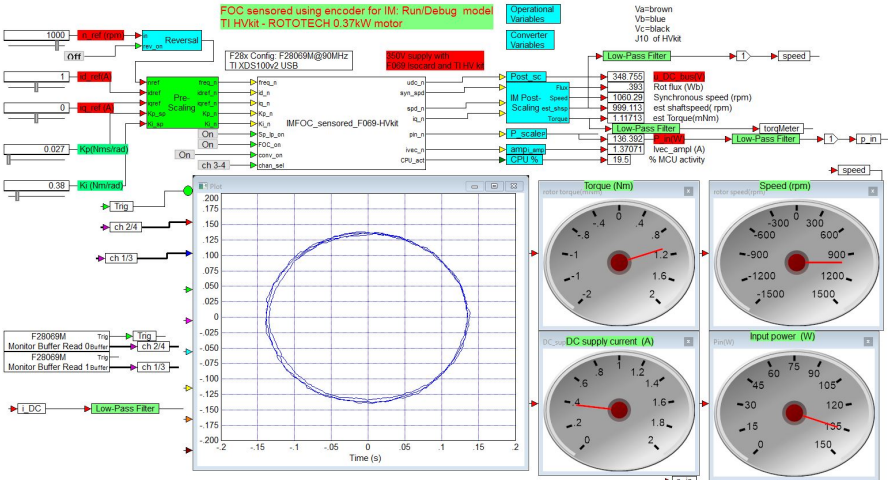


Fig. 6.44 HIL-Debug model, F069-HV kit connected to an induction machine, with field-oriented control using an encoder

The plot given in figure 6.44 displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit LPF voltage vector locus. The shaft speed

reference in figure 6.44 is set to 1000 rpm.
These results shows operation under FOC-speed control with the machine operating with a (hand applied) load. An instructive exercise for the user is to switch operation to FOC 'torque control' with the direct axis current set to produce rated rotor flux 0.4 Wb and quadrature current reference initially set to zero. Then hold the rotor by 'hand' and feel the effect of torque application when moving the quadrature reference slider.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 350 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

6.3 Sensored Field Oriented Control-IM with Dynamic Field Control (DFC)

So called 'dynamic field control' (DFC) refers to active use of the direct axis reference current to control the magnetic rotor flux level when operating with a partial mechanical load connected to the induction machine. The purpose of DFC is to reduce the input power requirements when operating with reduced mechanical load, as may be observed from figure 6.45.

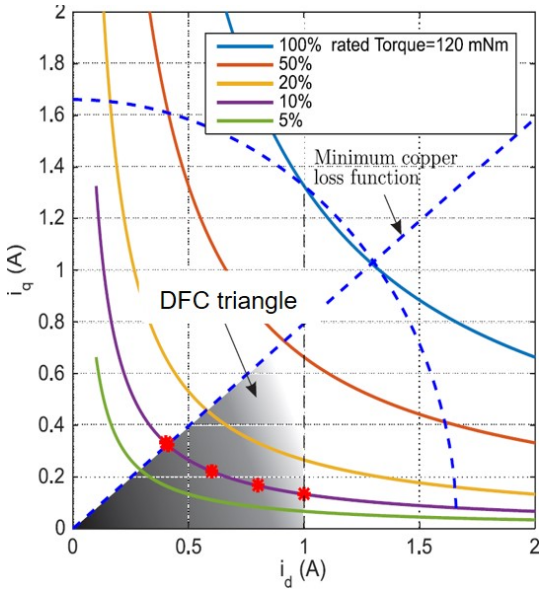


Fig. 6.45 DFC Operation of an IM drive with a field-oriented FOC/speed Controller

This figure shows steady FOC operation of the induction machine used in these examples for a range of load torque conditions. A 'DFC triangle' is apparent in figure 6.45, which is formed by the 'minimum copper loss' line and the 1 A rated direct axis value. The Gradient of the 'minimum copper loss function' is set by the user as a $^{i_d}/_{i_q}$ ratio, typically in the order of $1 \rightarrow 1.5$. For the magnetically linear case (with FOC active), this ratio can be expressed in terms of an angle $^{i_d}/_{i_q} = \cot \epsilon$ (which is the angle between the vertical axis and the 'minimum copper loss function') and current vector amplitude i_s. For the magnetically linear case the process of determining the optimum ϵ value that yields the highest shaft torque (for the lowest current amplitude, known as the MTPA point) makes use of expressions (6.7), (6.8) together with a constraint that the current vector locus is a circle with given value i_s as shown in expression (6.12), which also shows the direct/quadrature current values as function of the angle ϵ.

$$i_d = i_s \sin \epsilon \tag{6.12a}$$
$$i_q = i_s \cos \epsilon \tag{6.12b}$$
$$i_s^2 = i_d^2 + i_q^2 \tag{6.12c}$$

Substitution of (6.12a),(6.12b) into (6.8) gives:

$$T_{\mathrm{me}} = \frac{3}{2}\, p\, L_M\, i_s^2 \cos\epsilon \sin\epsilon \tag{6.13}$$

Partial differentiation of the shaft torque with respect to the angle ϵ gives:

$$\frac{\partial T_{\mathrm{me}}}{\partial \epsilon} = \left(\frac{3}{2}\, p\, L_M\, i_s^2\right)\left(1 - 2\,\sin^2\epsilon\right) \tag{6.14}$$

The optimum angle ϵ^{opt} can be found with the aid of (6.14) and using the condition $\frac{\partial T_{\mathrm{me}}}{\partial \epsilon} = 0$, which leads to expression:

$$\sin\epsilon^{\mathrm{opt}} = \frac{1}{\sqrt{2}} \tag{6.15}$$

Hence the theoretical optimum ratio i_d/i_q is simply 1. However, saturation of the magnetization curve $\psi_R\,(i_d)$ will compromise the relationship $L_M = \psi_R/i_d$ and the validity of expression (6.15). This implies that the ratio i_d/i_q, will increase, as mentioned above.
Note that operating with the condition $i_d = i_q$ implies that the direct axis current is changed as the load (and therefore the quadrature current) changes. However, rotor flux changes with NOT be immediate, as this is a function of the rotor time constant (L_M/R_R).
The hyperbola function (see figure 6.45) that represents 10% rated torque, shows a set of operating points starting at rated i_d and ends at the operating point on the 'minimum copper loss function'. This shows that under partial mechanical load it is beneficial to reduce the direct axis reference current in order to reduce the input power (and maximise the torque/current ratio MTPA) for the drive. Hence the purpose of DFC is simply to operate on the 'minimum copper loss' line, which is particularly useful WHEN operating under partial load conditions, i.e within the 'DFC triangle'.

6.3.1 IM Sensored Field Oriented Control with DFC-TI

The purpose of this section is to realize sensored Field-Oriented Control with 'dynamic field control' (DFC) of an Induction machine [25] using Texas Instruments hardware. The tool chain introduced in figure 2.45 is used as a

template for the ensuing subsections covering SIM→HIL, which is based on the `F0049-GANINV` setup.

6.3.1.1 IM Sensored Field Oriented Control wth DFC: F0049-GANINV:SIM

Reference program: `IM-DFC-FOC-F0049-GanINV-SIM.vsm`

This example considers the simulation of the IM motor drive in use, with fixed point FOC, DFC and speed controller as shown in figure 6.46

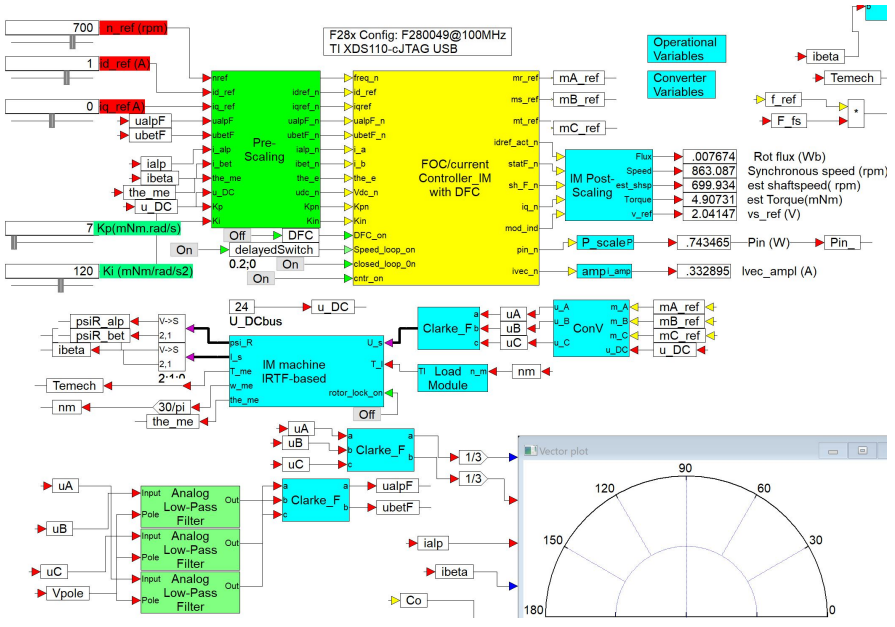


Fig. 6.46 Simulation of an induction drive with FOC-DFC/Speed Controller

The controller outputs are the three reference modulation indices `mA_ref`, `mB_ref`, `mC_ref`. These variables are connected to a 'CONV' module which multiplies the modulation indices by the term $u_{DC}/2$. Said module also has three limiters, that limit the voltage of the motor to $\pm u_{DC}/2$. A Forward Clarke transformation is again used to generate the voltage vector for the machine model. Outputs of the machine model are the currents i_α, i_β shaft torque T_{me} shaft speed n_{me} and shaft angle θ_{me}. The latter variable is used by the controller to calculate the appropriate rotor flux angle θ_{ref} (see equation (6.7b)) for the machine. In addition, the motor model generates the rotor flux variables $\psi_{R\alpha}$, $\psi_{R\beta}$, which are used to show the scaled rotor vector in a

polar plot. A set of sliders is used to define the current reference vector, in terms of its direct and quadrature components i_d^{ref}, i_q^{ref}, and rotational speed $n_{\text{me}}^{\text{ref}}$. A 'pre-scaling' module (as discussed on section 1.4.2) has been added to convert the floating point variables required for the controller to fixed point format. In addition, two sliders are provided, which allows the user to set the proportional K_p^{sp} and integral K_i^{sp} speed gains of the controller. Note that said gains are shown in (milli)Nms/rad and (milli)Nm/rad respectively. A guideline for choosing these parameters is given in section 2.1.2.
The dialogue box 'Operational Variables' for this example, as shown in figure 6.47, contain all the user settings required.

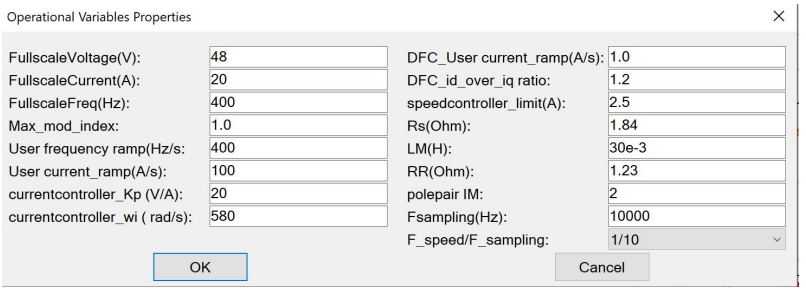


Fig. 6.47 Simulation: Dialogue box 'Operational Variables'

This dialogue box is again used to assign full scale values for the voltage, current and frequency. In addition, the current controller gain and bandwidth are also assigned here. These values are based on the motor in use, as was discussed in section 2.1.4. A dialog box entry for the maximum modulation index value and speed controller limit value `speedcontroller_limit(A)` are also shown. The latter value is the maximum (absolute) quadrature current value which the speed controller is assigned to provide. Note that the maximum modulation index value corresponds to a circular limit in the α, β coordinate system, of which the radius is set by said variable. Also assigned in this box are the rate of variable change for current(s) and frequency, i.e. introduced as a safety measure to avoid erratic slider action from causing over-currents in the drive. In addition, this dialogue box assigns the motor pole pair number and required motor parameters (four parameter model representation (see figure 2.25) of the induction machine in use. Furthermore, the sample frequency used for the speed controller module is set in terms of a ratio relative to the controller sample frequency, which is also indicated in this dialogue box for convenience. Note that the actual controller sampling rate is set in the Embed 'Systems property dialogue box'.

With respect to DFC operation two variables have been introduced namely:

- `DFC_id_over_iq ratio`: which defines the gradient of the 'minimum copper loss function, see figure 6.45 and set by the user.
- `DFC User current ramp (A/s)`: which is rate of current decay used by the DFC controller to reach the mimimum power point , when DFC is active. The rate increase is defined by the standard current ramp `User current_ramp (A/s)`.

Moving one level lower into the 'FOC/Current Controller IM with DFC' module reveals a set of modules given in figure 6.48.

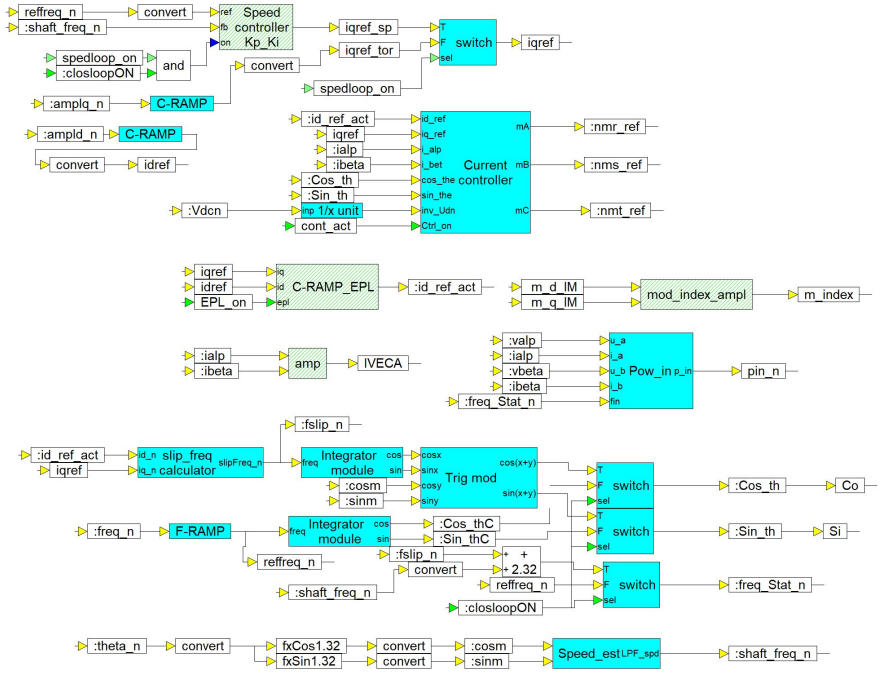


Fig. 6.48 Fixed point representation of the IM FOC-DFC/Speed Controller

Synchronous model based current control is undertaken by a 'Current Controller' module, which has as inputs the per unit quadrature and direct axis reference value, measured α, β currents, Cos/Sin inputs of synchronous reference frame and inverse DC bus voltage. The latter is generated by the '1/x ' module, which generates the per unit inverse DC bus voltage using a numerical routine, as discussed in previous examples. A 'switch' module, activated by the `closloopON` signal (active if the shaft encoder is required) is used to select which Cos/Sine variables are to be used by the current controller module. When this switch is `OFF`, operation of the drive is under open loop current & frequency control, which is a good initial step to verify operation

of the current controller. When `closloopON` is ON a 'slip freq calculator' and 'integrator' input are used in combination with a `Trig module` to generate the Cos/Sin signals for the current controller. The function of said module'is to generate the variables $\cos(\theta_{\text{slp}} + \theta_m)$, $\sin(\theta_{\text{slp}} + \theta_m)$. Inputs to this module are the cos/sine of the slip frequency and the Cos/Sine of the electrical shaft angle `cosm`, `sinm`.
Inputs to the speed controller are the (rate limited) reference frequency `reffreq_n` and
estimated shaft frequency (from the 'speed est' unit) `shaft_freq_n`. Output of the speed controller module (when enabled) is the per unit quadrature reference current `iqref_sp`. Note that the speed controller module is shown 'hatched green', which implies that its sample rate is different to the remaining part of the controller. In this case the sample frequency for the speed controller is set to 1 kHz. For less time demanding activities it is prudent to use a lower sampling frequency, which serves to avoid MCU overrun errors that can occur when insufficient time is available to execute all the required function for a given ADC frequency. A 'switch' module with output `iqref` determines whether the quadrature reference current for the current controller is provided by the speed controller or the user i_q^{ref} slider. The switch in question is controlled by the `speedloop_on` signal (active when closed speed-loop control is required, 'off' implies operation in 'torque mode'). A set of 'Rate Limiter' modules, entitled 'C-ramp' (sets the current ramp rate) and 'F-ramp' (sets the frequency ramp rate), have been introduced in figure 6.48 to avoid erratic changes to the user input variables (as actually used by the controller), which may damage the converter or motor. The rate of variable change is specified in the 'Operational Variables' dialogue box (see figure 6.47).

Added to this controller is a sub-module `C-RAMP_EPL`, which executes the DFC control algorithm and is controlled by the logic variable `EPL_on` which, in turn, is controlled by user button `DFC_on`. Output of this sub module is the per unit direct axis current `id_ref_act`, which ramps down said current using the user defined DFC ramp rate (when DFC is active).

An example of drive operation is shown in figure 6.49. The sampling time T_s as defined in the program pull-down menu is set to 0.2 s, i.e a 10 kHz sampling frequency is used for the discrete controller model. A quadratic load torque/speed characteristic is used which provides a torque of 10 mNm at 1000 rpm, hence partial load operation.
The top sub-plot (see figure 6.49), shows the motor phase current and phase voltage using sensored, field-oriented, speed control (initiated at $t \approx 0.2$ s) with DFC active at time mark 0.8 s, which is activated at $t \approx 0.1$ s. The delay in speed control operation, is purposely done to ensure that the induction machine flux level is at its rated level 30 mWb, before speed and/or DFC control is undertaken. In this case the simulation was run over a 2 s time interval, with a motor inertia J=$86.6 \cdot 10^{-6}$ kgm^2 (taken to be the estimated

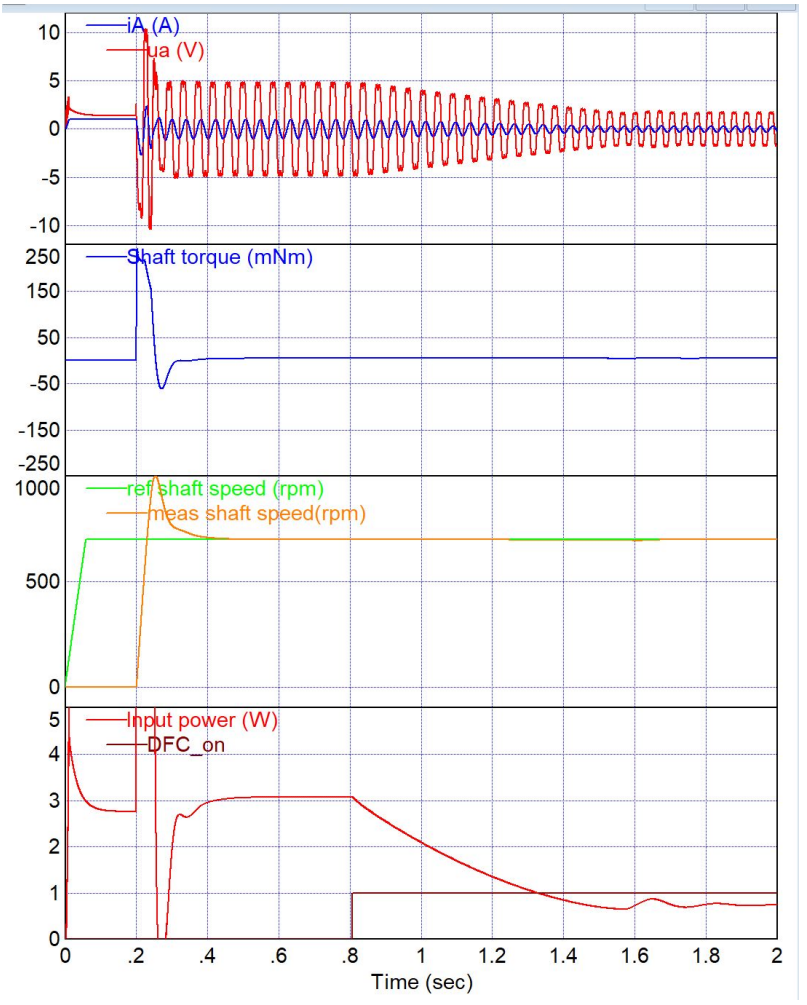


Fig. 6.49 Simulation: Typical waveforms obtained using a FOC-DFC/speed IM controller

combined inertia of the PM/IM motor unit). The third lower sub-plot, shows the actual and reference shaft speed for a given frequency ramp setting (as set in the dialogue box). Also shown in this sub-plot is the actual shaft torque of the machine over the 2.0 s simulation interval. The lower sub-plot (see figure 6.49), shows the impact of using DFC on the motor input power.
Some interesting observations can be made from the results shown in figure 6.49 during operation namely:

- the maximum phase current is determined by the maximum $i_q^{\max}$ value set in the speed controller and the chosen direct axis reference current value.

Hence with $i_d^{\text{ref}} = 1.0$, $i_q^{\max} = 2.5$ the maximum peak phase current will be equal to $i_s^{\max} = \sqrt{1.0^2 + 2.5^2}$=2.69 A.

- The phase current will be determined by the selected direct axis reference current value and quadrature current required for the load, when the shaft speed is constant.
- The maximum shaft torque during startup can be written as $1.5\,p\,L_M\,i_d^{\text{ref}}\,i_q^{\max}$, which is equal to 225 mNm in this case, given the chosen initial direct axis reference current setting of 1.0 A.
- With DFC active, the steady-state direct axis current will be determined by the quadrature current value in use (defined by the mechanical load) and the variable `DFC_id_over_iq ratio`
- The rate of current decay when DFC is active is defined by the variable `DFC User current ramp (A/s)` set to 1 A/s in this example.

In the sequel to this section a vector plot is shown of the drive in question operating under steady state conditions, with a constant speed of n^{ref}=700 rpm and arbitrarily chosen mechanical load of 4.6 mNm at time mark ≈2 s.

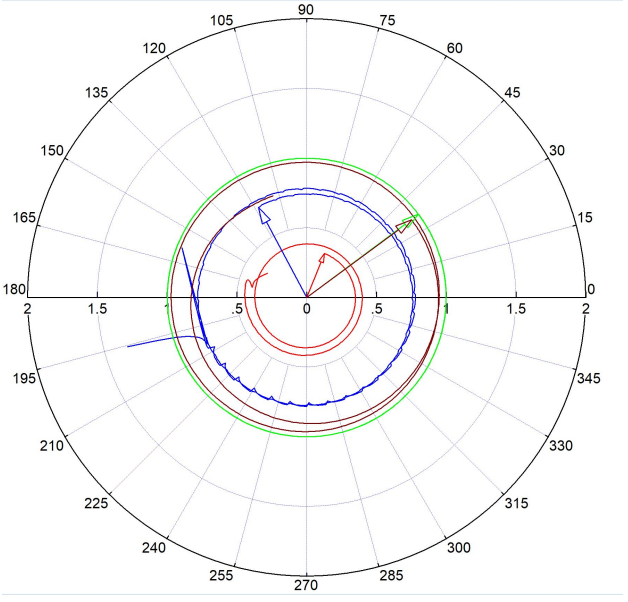


Fig. 6.50 Simulation: Vector plot obtained using a FOC/Speed controller with current $\vec{i}_s$ ('red'), voltage vector $\vec{u}_s/3$ ('blue), rotor flux vector $\vec{\psi}_R \times 100$ ('brown') and unity d-axis vector ('green')

Said vector plot (see figure 6.50), shows the current vector $\vec{i}_s$ and voltage vector $\vec{u}_s$, whereby the latter is scaled by a factor 3. In addition, a unity

'd-axis' vector is introduced to show the orientation of the d,q reference frame, where the positive d-axis is displayed as a 'green' vector, which is aligned with the rotor flux vector. Its magnitude is set by the product of the direct axis current and magnetizing inductance L_M. In this example, a current i_q value is generated, given that the machine is producing torque to match the load torque applied. Finally, the flux rotor vector (scaled by a factor 100) generated by the IRTF motor model is also added and is aligned with the controller 'd-axis'. With DFC active, as is the case here, the rotor flux is significantly reduced when compared to the situation when DFC is NOT used. Note that the orientation of the current vector relative to the controller d-axis is now controlled by the user variable `DFC_id_over_iq ratio`, as may be observed from the vector plot.

6.3.1.2 IM Sensored FOC with DFC and Enc-F0049-GANINV: HIL

Reference files:

- `IMFOC_DFC_enc_F0049-GANINV-compile.vsm`, for HIL-compile
- `IMFOC_DFC_enc_F0049-GANINV-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), shown in figure 6.30, represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F280049` module), which is connected to a induction machine[25]. In addition, this motor is also mechanically connected a PM machine[2] because use is made of its encoder for sensored IM FOC operation.

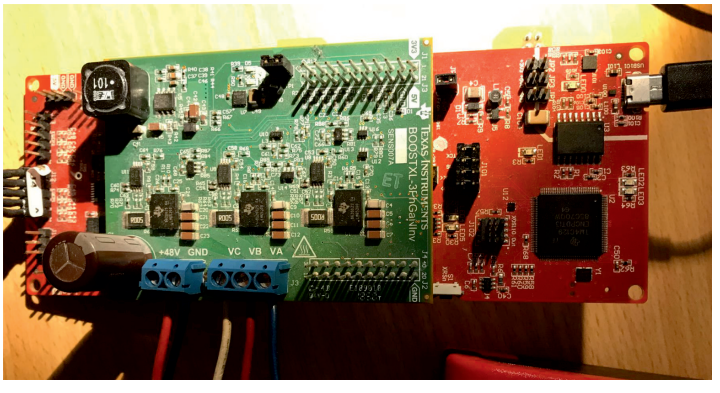


Fig. 6.51 MCU-converter setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the

converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 6.52 shows the 'FOC-DFC sensored and C&F control' module, which must be compiled to generate an out-file.

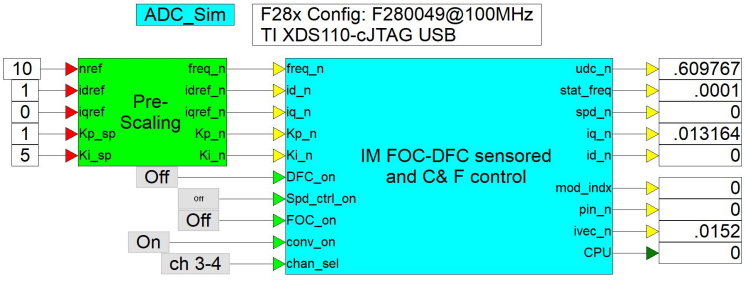


Fig. 6.52 HIL-Compile model of an induction machine drive with field-oriented control, DFC and encoder

Inputs, shown in the stage as constants (for readability purposes), for this controller module are the user direct/quadrature axis reference currents `id_n`, `iq_n`, and the reference shaft frequency `freq_n`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller.
Five logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other three are used to select field-oriented control (FOC) operation `FOC_on`, speed operation under FOC `Spd_ctrl_on` and `DFC_on` control, as discussed earlier (see section 6.3.1.1).
Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- A per unit synchronous speed estimate `stat_freq` generated via the controller.
- A per unit shaft speed estimate `sp_n` generated via the encoder.
- The measured per unit quadrature current `iq_n`, used to generate a shaft torque estimate.
- The reference per unit direct axis current `id_n`, used to generate a rotor flux estimate.
- The modulation index amplitude `mod_indx` used to calculate the reference voltage vector amplitude.

- A per unit estimate of the motor input power `pin_n`. This is also useful as an estimate for the drive input power (but this ignored the converter losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx p_{\mathrm{in}}/u_{\mathrm{DC}}$.
- The measured per unit current vector amplitude `ivec_n`.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 6.52 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 6.53.

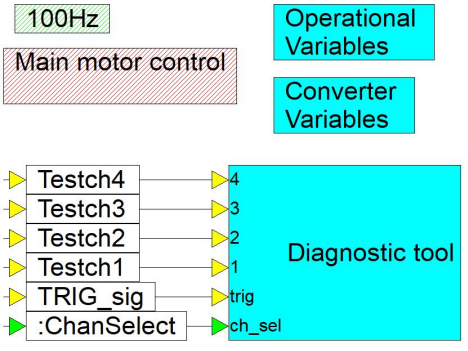


Fig. 6.53 One level down into the IM FOC sensored Controller module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current(s)/frequency, as discussed earlier (see figure 6.47). The 'Converter Variables' dialogue box (see figure 6.34), defines the scaling used by the Analogue to Digital converter (as discussed earlier, see equation (4.2)).
This dialogue box also shows the low-pass filter corner frequency used by the converter for measured the half-bridge voltages (see figure 4.12).
A 'Diagnostic tool' module is used to buffer two variables, which can subse-

quently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 6.53 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 6.54. Of these, the 'FOC/speed controller IM with DFC' has been discussed earlier (see figure 6.48), where it was used in phase: SIM of the development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F280049` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present, which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after every second EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.
The `ADC-PWM-LaunchXL-28004x BOOSTXL-3phGANINV` unit is used generate the PWM signals for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively. ADC offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example (see figure 6.55), and its task is to execute the `XXX.out` file

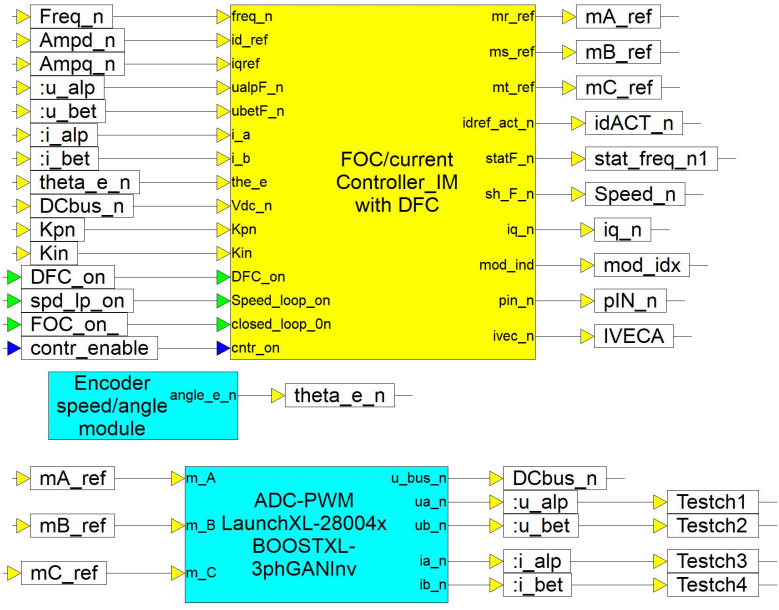


Fig. 6.54 One level down into the 'Main motor controller' module

compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. Two sliders are used to set the direct axis current reference value and the electrical frequency . Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.
The plot given in figure 6.55 displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit LPF voltage vector locus. The shaft speed reference in figure 6.55 is set to 1000 rpm.
These results shows operation under FOC-speed & DFC control with the machine operating under no-load conditions. When DFC is activated the per unit current locus changes from $0.05 \rightarrow 0.035$, at a DFC decay rate set (purposely slow to observe operation) to 0.1 A/s. The current vector locus with `DFC off` was given earlier (see figure 6.36), and clearly shows the importance of DFC control, when there is a need to reduce input power requirements for partially mechanically loaded drives.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with

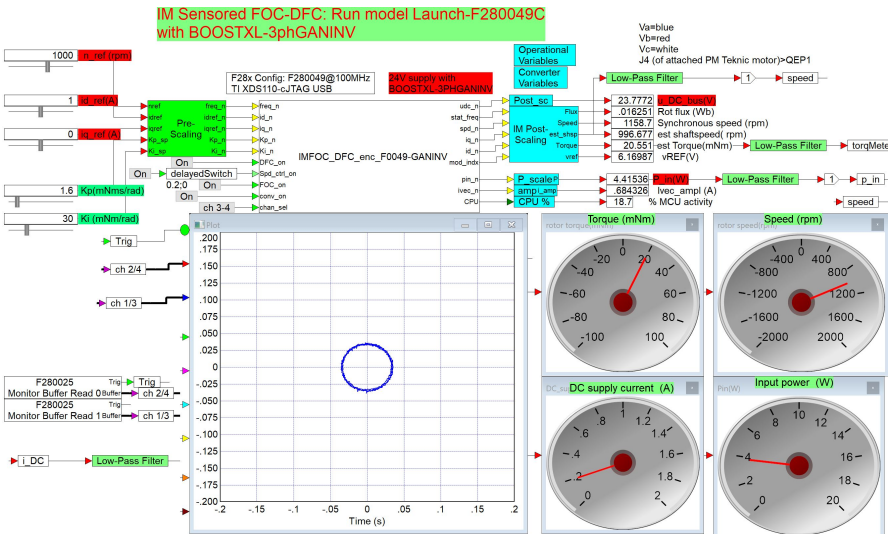


Fig. 6.55 HIL-Debug model, F0049-GANINV of an induction machine drive with field-oriented control using an encoder and DFC active

live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

6.4 Sensorless Field Oriented Control three-phase IM

The term 'sensorless' specifically refers to the exclusion of a shaft speed or encoder sensor, normally used to achieve field-oriented control of a IM drive. When using an encoder for IM, the aim is to identify the shaft speed (or shaft angle) [13] and then calculate the synchronous frequency ω_s and angle $\theta = \omega_s\, t$ (see figure 6.56), as discussed earlier (see section 6.2).

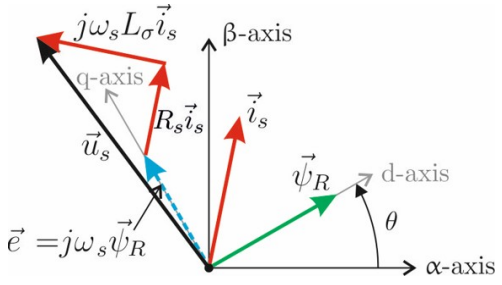


Fig. 6.56 Terminal vector plot: FOC operation induction machine drives

Encoderless FOC control is principally based on identification of the EMF vector $\vec{e}$ in terms of its amplitude and orientation in the α, β coordinate system. Once this is established, the rotor flux is estimated by integrating said EMF vector, using:

$$\vec{\psi}_R^{\text{est}} = \frac{\vec{e}}{j\omega_s} \tag{6.16}$$

Sensorless IM field-oriented control is then established, whereby the direct axis current component (which is tied to the estimated rotor flux vector, hence in alignment) determines the magnitude of said flux vector.

Equation (6.16) shows an important issue with respect to rotor flux estimation. When the electrical frequency moves from $\omega_s \to 0$ the EMF amplitude will likewise move: $|\vec{e}| \to 0$. Which implies that expression (6.16) becomes undefined at zero synchronous frequency, which for an induction machine typically differs from the shaft speed frequency (which depends on the applied mechanical load). Hence, an induction machine with load can, for example, operate under locked rotor conditions, because the synchronous speed will then be unequal to zero.

Sensorless control for IM makes use of the 'synchronous' d, q reference frame and the shaft speed of the machine is calculated using expression (6.17),

$$\omega_{\text{me}} = \left(\omega_s - \frac{i_q R_R}{\psi_R}\right) \frac{1}{p} \tag{6.17}$$

where ω_{me} is the shaft speed in rad/s. This expression requires access to the rotor resistance value R_R. Vice-versa equation (6.17) can be used to estimate this parameter R_R for the case where the rotor is purposely locked (which means $\omega_{\text{me}} = 0$), as will be discussed at a later stage. Expression (6.17) requires access to the rotor flux magnitude ψ_R that is generated by the encoderless algorithm used.
For IM FOC drives the flux is controlled by the direct axis current i_d^{ref} an example of this relationship, known as the magnetizing curve, is given in figure 6.57.

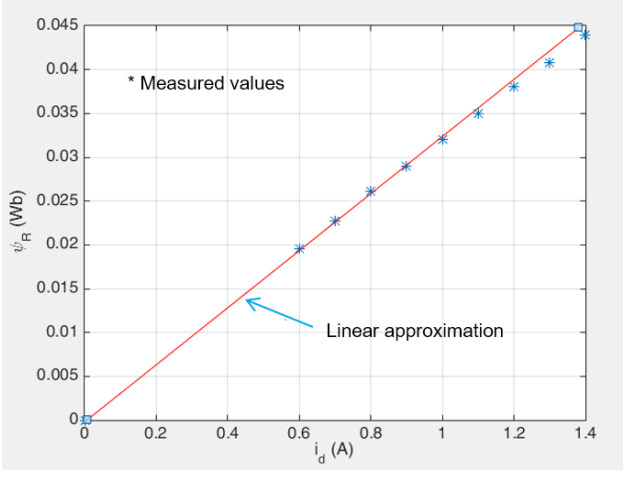


Fig. 6.57 Typical magnetizing characteristic of an induction machine

Added to figure 6.57 is a linear function, of which the gradient represents the magnetizing inductance L_M of the machine. If saturation is limited, as is the case for the machine shown, the rotor flux can be calculated using $\psi_R = L_M\, i_d$ as shown earlier (see figure 6.16). However, for encoderless operation this linear approximation is NOT required, as the rotor flux is directly estimated using the AFT®or FAST®algorithm.

To achieve 'good' encoderless control the following guidelines should be adhered to namely:

- Accurate measurement of the terminal voltage vector $\vec{v}_s$. Here use is made of the low-pass filtered converter voltages which are then Clarke transformed. The use of low-pass filtered voltages implies the need for accurate knowledge of the LPF corner frequency implemented in the converter, as this information must also be passed to the sensorless algorithm.
- Accurate measurement of the current vector $\vec{i}_s$. When using current shunts in the lower converter switches, current measurements when operating at high modulation index values, e.g at high speeds, may be compromised and this can affect EMF estimation. Using phase 'in-line' current shunts/transducers is therefore preferred for sensorless FOC.
- Compensation of ADC offsets for voltage and current measurement.
- Optimum utilization of the typically 12bit ADC converter(s), which in effect means matching the the ADC-fullscale voltage/current settings of the drive to the application as discussed earlier, see section (4.1.1.3).
- Accurate stator resistance estimation: as the synchronous speed reduces the estimated EMF vector is dominated by the term $\left(\vec{v}_s - R_s\,\vec{i}_s\right)$. In this context an important figure of merit was introduced earlier and referred to as the 'EMF threshold speed' n_e (see equation (2.12a). This (for an IM drive) is the synchronous speed at which the EMF amplitude equals the voltage drop across the resistance and this can also be expressed as a ratio $\left(\frac{n_e}{T_{\rm me}}\right)$, where $T_{\rm me}$ represents the shaft torque and may be calculated using:

$$\frac{n_{\rm e}}{T_{\rm me}} = \frac{20\,R_s}{\pi\;p^2\psi_R^2}\,(\text{rpm/Nm}) \tag{6.18}$$

  Hence accurate voltage/current measurements AND accurate stator resistance value is needed, particularly at start-up of the drive. Which is why resistance estimation, using DC current injection is preferred at that instance.
- Accurate estimation of the leakage inductance L_σ: as synchronous speed increases (and load) the voltage across the inductance becomes an important term for EMF estimation as may be observed from figure 6.56. The importance of this term is dependent on the ratio quadrature current/short circuit current, as discussed earlier (see equation (2.10)). Note that for induction machines an 'equivalent short-circuit' current is introduced, defined as $i_{\rm sc}^{\rm eq} = \psi_R/L_\sigma$.
- Good knowledge of the machine rotor flux: a sensorless algorithm requires access to a good estimate of the rotor flux, which for IM drives depends on the choice of reference direct axis current. This estimate may be found by operating under current & frequency control conditions (see figure 6.37).

The benefits of using encoderless FOC (when compared to using a shaft sensor) are as follows:

- No need for adding a shaft sensor to the electrical machine, which reduces the overall space needed for the drive and reduces the amount of cabling to the machine (no sensor cable, just the three phase wires).
- In applications where both shaft ends of the motor are required the use of a shaft sensor can be problematic.
- For applications where reliability of drive operation is critical, the use of encoderless FOC is often used as a 'backup'. For example, sensor failures can then be bypassed by moving to sensorless FOC, hence ensuring continuity of operation.
- Sensorless FOC functions as a high resolution encoder, which improves controllability of the drive.
- Cost issue: the removal of the shaft sensor reduces the overall cost of the drive.

The advantages of using sensorless FOC control for IM drives must be weighed up against the issues mentioned in the 'guidelines' above. However, it is important to note that shaft speed accuracy for IM sensorless drives is compromised, due to the need to have good knowledge of the rotor resistance, as may be deduced from equation (6.17). This parameter is affected by the operating conditions, i.e heating up of the rotor. Furthermore, skin-effect (high slip conditions) can also affect R_R, which in turn implies that the estimated shaft speed may not be exactly the same as the actual shaft speed.

6.4.1 Sensorless Field Oriented Control of IM using FAST®-TI

The examples linked to Texas Instruments will be covered using the development tool-chain components: PIL→COSIM→HIL, based on the use of the Piccolo [23] series: F28027 and F28069 Launchpad's. These MCU's have access to the Texas Instruments software InstaSPIN®, which contains their classified sensorless algorithm FAST®. Access to FAST®is by way of an Embed DLL, which implies that simulation is not an option when using this algorithm. Initially the tool-chain components presented in the ensuing subsections will be based on the use of the F28069-GANINV drive set-up with a induction machine [25], but other HIL examples will also be given.

6.4.1.1 IM FAST FOC-F069-GANINV:PIL

Reference files:

- `FOC_IM_FAST_F069-GANINV-PIL.vsm`
- `IM_FAST-PIL_mod.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F28069, which contains the DLL used to communicate with FAST. This implies the use a target execution module for file: `IM-FOC_FAST-PIL.out` as indicated in figure 6.58.

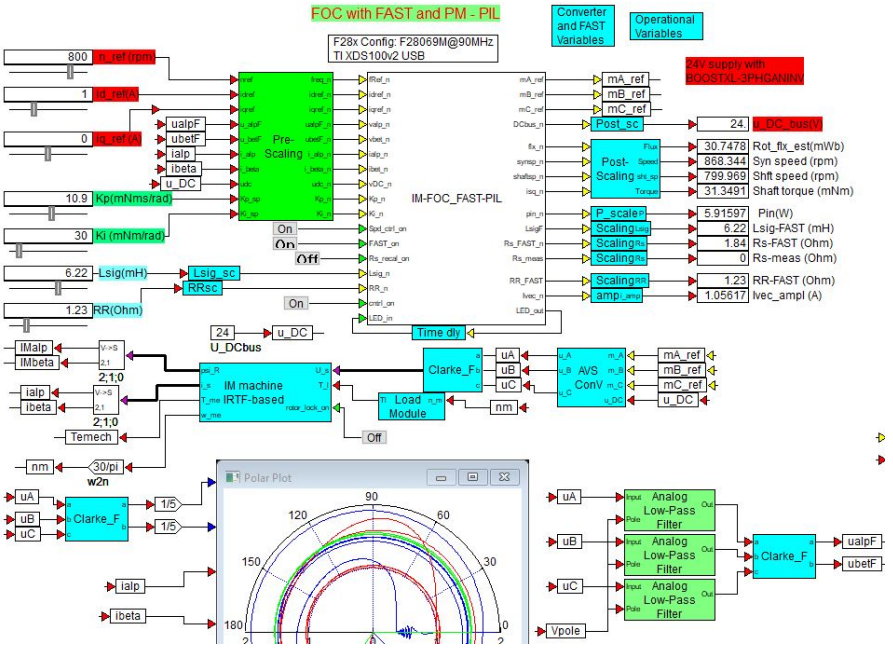


Fig. 6.58 Field-oriented torque/speed control for IM using FAST®:PIL operation

The target interface module that executes the out-file, generates the modulation indices required for the converter/motor and these components are not affected by use of the PIL. Generation of the out file is based on the PIL module (see figure 6.59), which must be compiled using the command: `toolbox>codegen`.
Moving one level into the PIL module `IM-FOC-FAST-PIL` shows a 'Drive Controller' (see figure 6.60).
Added to said controller is a module `LED_ctrl`, which is used to flash an LED

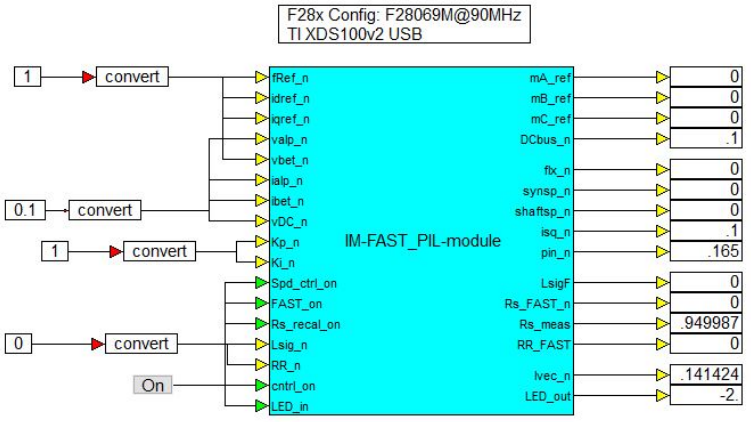


Fig. 6.59 Module:'IM-FOC-FAST-PIL'

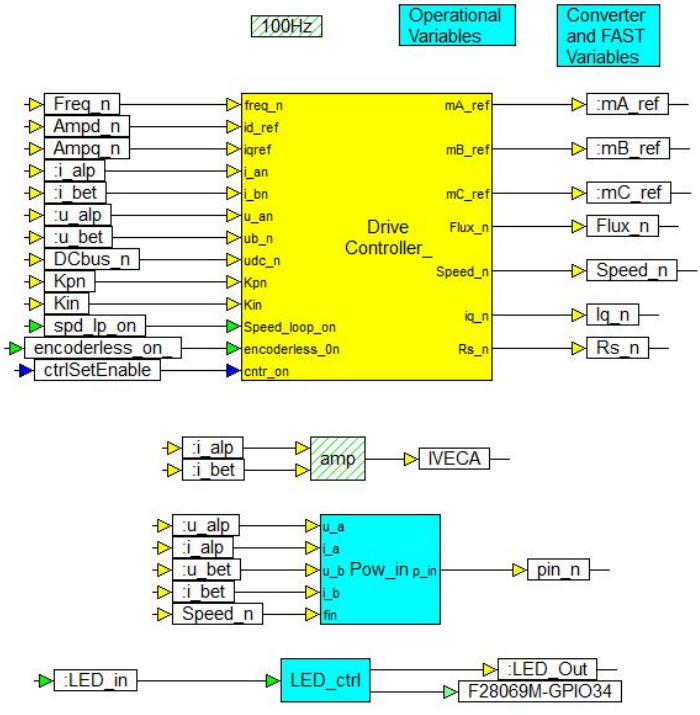


Fig. 6.60 Content PIL module, Field-oriented torque/speed control with FAST®

on the MCU board to show that the PIL control is operational. Furthermore, the modules responsible for input power measurement `Pow_in` and current vector amplitude `amp` are located outside the 'Drive Controller'.
The content of the 'Drive Controller' (see figure 6.61), shows the `Current controller` and `Speed Controller` as introduced earlier (see section 4.3.1.1).

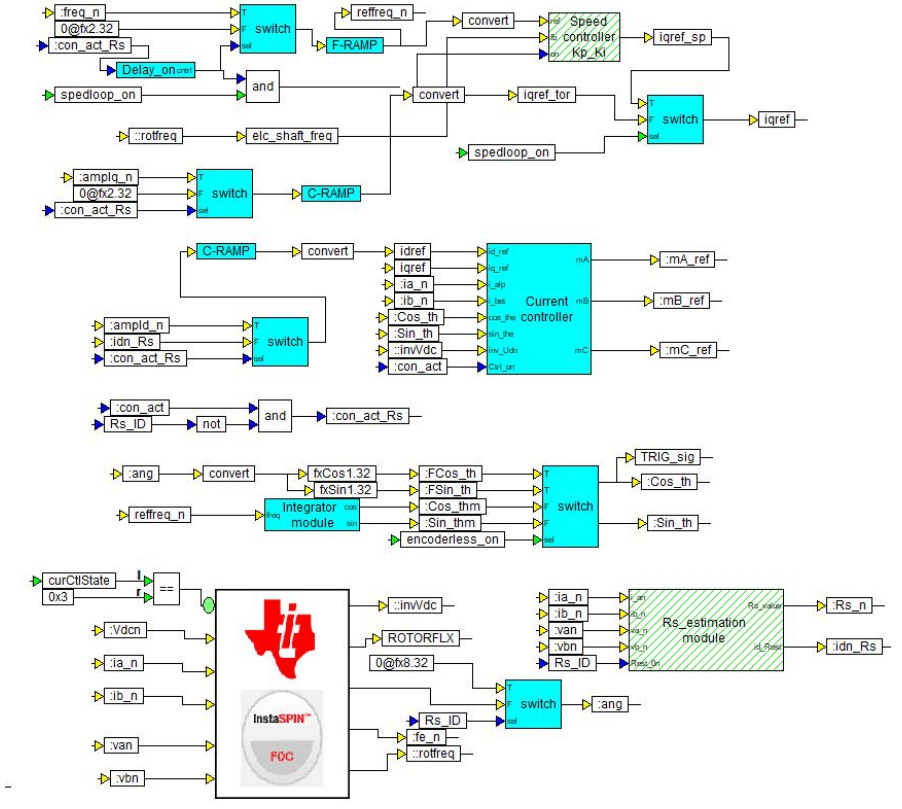


Fig. 6.61 Content 'Drive Controller' module

Note that the per unit inverse bus voltage variable `invVdc` (as required for the current controller), is now generated by the FAST®module `InstaSPIN FOC`. Said FAST®module provides per unit estimates, of the rotor flux amplitude `ROTORFLX`, rotor flux angle `ang`, electrical shaft frequency `rotfreq` and synchronous frequency `fe_n`. This module can also provide an estimate for the torque `eleTT`, but that is not used here, as it is directly calculated using the rotor flux estimate together with the per unit measured quadrature current variable `iq` generated by the 'Current controller'.
Inputs to the FAST®module are the per unit measured LPF voltages and

currents, which are Clarke converted to stationary α, β coordinates. In addition, a variable `CurCTLstate` (generated in the `100 Hz` module) activates the FAST®module, which is also provided with the per unit measured DC bus voltage `Vdcn`. Also shown in figure 6.61 is a module `Rs_estimation`, which (when activated by the user with variable `Rs_ID`), estimates the stator resistance by injecting a direct axis current (amplitude set in the 'Operational Variables' dialogue box). Note that it is prudent to choose a DC injection current (for Rs estimation) that matches the direct axis current reference used for rotor flux control. This implies that during resistance measurement the rotor flux will equal the value used by FAST®(this assumes that a good estimate for the rotor flux is known that corresponds to the direct axis current reference) . When stator resistance estimation is active, the angle output of the FAST®module is held to zero.
When FAST®is not used (encoderless off) the logic variable `encoderless_on` is zero and this implies open loop current&frequency control is active. However, the FAST®module will also remain active under these conditions, hence an effective mode to test operation of this module.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational.

Operation of the drive is at a shaft speed of 800 rpm (see figure 6.62), with speed control active and the load controls the quadrature current amplitude (once steady state operation is achieved), as mentioned earlier. FAST®operation starts at 0.32 s and the torque is initially limited by the 2.5 A quadrature current setting for the speed controller.
The vector plot (see figure 6.63), which corresponds to time mark 0.6 s, is interesting, as it shows the scaled (by a factor 50) rotor flux vector ('green'), current vector ('red') and scaled (by a factor 1/5) terminal voltage vector ('blue').
A key observation from this vector plot is the orientation of the current flux vector relative to the rotor flux vector. In this example the direct axis current setting is 1 A and the quadrature current is defined by the load, as speed control is active.

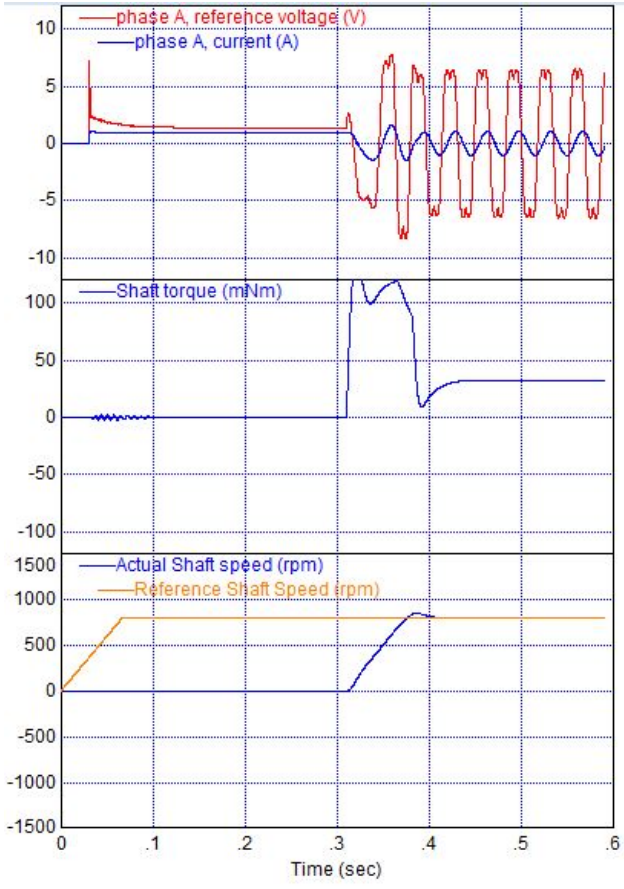


Fig. 6.62 PIL Result: FOC sensorless operation IM drive, using FAST®with speed control active.

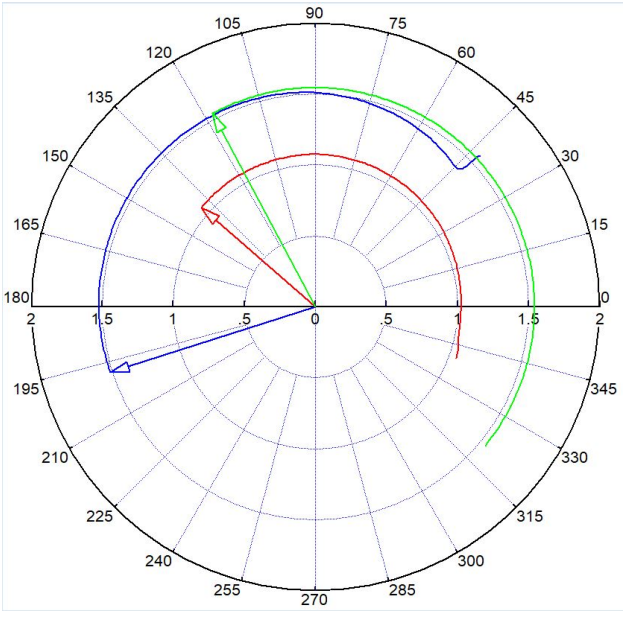


Fig. 6.63 PIL result, IM drive operation, sensorless FOC-speed control active:vector plot

6.4.1.2 IM FAST FOC-F069-GANINV: COSIM

Reference files:

- `IM-FOC-FAST-F069-GANINV-COSIM.vsm`
- `IM_FAST-PIL_mod.vsm`
- `GAN_INVv13.psimsch`
- `IM motor model_v1.psimsch`
- `IM-param_file.txt`

To evaluate operation of a controller with a switching converter, in this case the `GANINV`, use can be made of a `PSIMCoupler` module, as indicated in figure 6.64. This approach requires the use of PSIM together with Embed, which is referred to as 'co-simulation'. Furthermore, a 'processor in the loop' (PIL) approach is needed, given the use of a classified sensorless control algorithm FAST®.

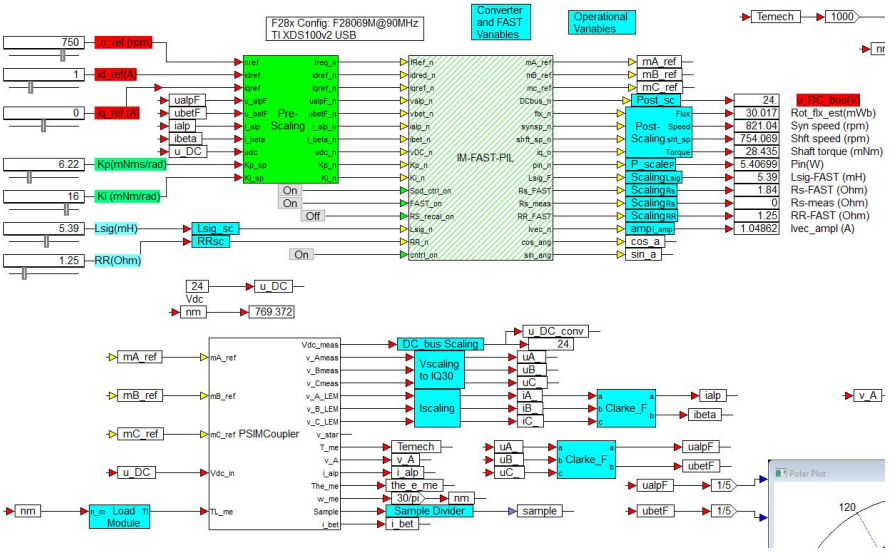


Fig. 6.64 PSIMcosim based IM drive, Field-oriented torque/speed control using FAST®

The `IM-FAST-PIL` module is shown hatched in this case, as its sampling frequency is locally set to 10 kHz, whereas the 'system' frequency (set in the 'system properties' dialogue box) is set to 20 MHz. The latter is much higher, given the need to accurately represent 10 kHz voltage switching waveforms generated by the PWM unit of the converter. The content of the `IM-FAST-PIL` module was introduced earlier (see figure 6.60). However, the current and

voltage measurements now make use of the signal processing circuitry connected to the half-bridges.
In this case the current transducers are in the converter phases (see figure 6.26). After sampling and signal-processing in the converter, these variables appear as: `iA_`, `iB_`, `iC_` in figure 6.64. The low pass filtered voltages (which removes the PWM component) of the three half-bridges (also present in figure 6.64) and these are scaled and sampled by the module `Vscaling to IQ30` to actual values: `uA_`, `uB_` and `uC_`.
The 'Converter and FAST Variables' dialogue box (see figure 6.64), defines the scaling used by the Analogue to Digital converter, which has a maximum input value of 3.3 V, and is directly linked to the signal processing undertaken for voltages and currents as discussed above. For example, the 'ADC-fullscale-voltage' value is the theoretical peak-to peak input voltage which must be applied to the low-pass filter network (see figure 4.12), to ensure that the ADC input level varies between $0 \rightarrow 3.3$V.
The current transducer voltages: `v_A_LEM`, `v_B_LEM`, `v_C_LEM`, are sampled by the `Iscaling` module (see figure 6.64), as mentioned above. When dealing with the actual MCU, the ADC converter also takes over this sampling process.
Apart from the measured voltage/currents generated by the converter, the PSIM coupler unit also provides the actual motor model current vector: `i_alp`, `i_bet` (in stationary coordinates), phase 'A' voltage and shaft speed, which are presented by the variables `v_A`, `i_alp`, `i_bet` and `n_m` (after conversion to rpm) respectively (see figure 6.64).
The PSIM induction machine model `IM` used in the coupler module (see figure 6.27), is based on the IRTF model discussed earlier (see figure 2.25).

Results achieved with the COSIM approach are shown in figure 6.65, using a quadratic load-speed characteristic, which generates 50 mNm at a shaft speed of 1000 rpm, with speed control active.
Indicated in said figure is the phase 'A' voltage waveform together with the reference voltage, phase current, shaft torque and shaft speed. Note that FAST®is activated around time mark 0.03 s.
Co-simulation, as discussed in this sub-section, is instructive for understanding the impact of using a switching converter on drive operation, e.g to assess the impact of shaft torque ripple due to PWM.

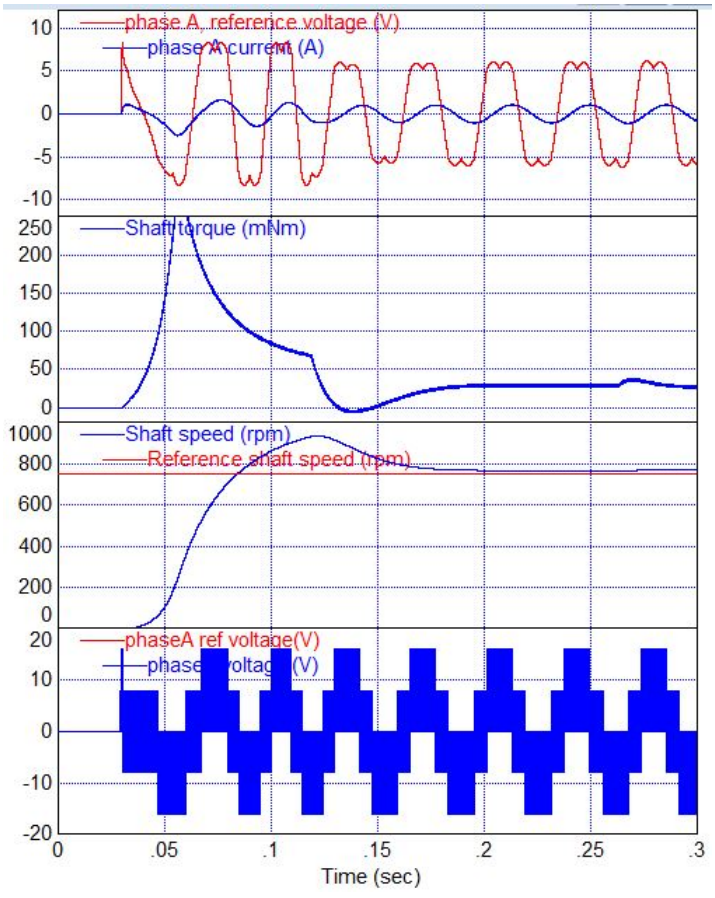


Fig. 6.65 Result COSIM, IM Field-oriented torque/speed control using FAST®

6.4.1.3 IM Sensorless FOC with FAST-F069-GANINV: HIL

Reference files:

- `IM_FAST_F069-GANINV-compile.vsm`, for HIL-compile
- `IM_FAST_F069-GANINV-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), shown in figure 6.66, represents the `BOOSTXL-GANINV` converter (located on the `LAUNCHXL-F28069` module), which is connected to an induction machine[25].

Fig. 6.66 F28069-GANINV setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches.
A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 6.67 shows the 'IM-FOC FAST' module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct/quadrature axis reference currents `id`, `iq` and the reference shaft speed `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller. Furthermore, two inputs `Lsig_pu`,`RR_pu` are provide to allow the user to set the leakage inductance L_σ and rotor resistance R_R (via sliders) respectively.
Five logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be

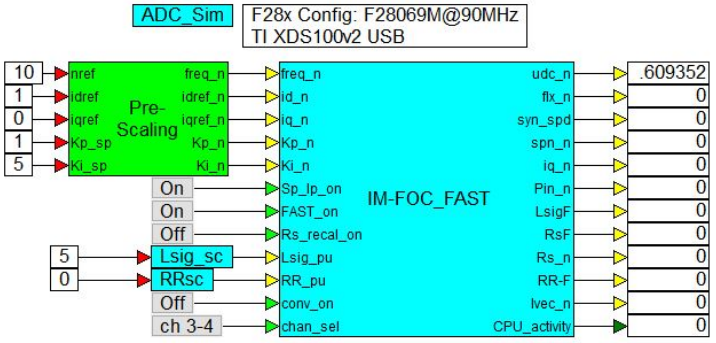


Fig. 6.67 HIL-Compile model of a Induction machine drive with field-oriented control and FAST®

viewed. The other three are used to select field-oriented control (FOC) operation with FAST® `FAST_on`, speed control operation with FOC `Sp_lp_on` and `RS_recal_on`, as discussed earlier (see section 6.4.1.1). The function: `RS_recal_on` estimates the stator resistance at start-up (when the button `conv_on` is activated).
Outputs of the Controller module are:

- The per unit DC bus voltage, `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- The per unit rotor flux estimate, `flx_n`.
- A per unit synchronous speed estimate, `syn_spd`.
- A per unit shaft speed estimate, `sp_n` .
- The measured per unit quadrature, current `iq_n`, used to estimate the shaft torque.
- A per unit estimate of the motor input power, `pin_n`. This is also useful as an estimate for the drive input power (but this ignored the converter losses) and can in turn be used to obtain an estimate of the DC bus current using, $i_{\mathrm{DC}} \approx {}^{p_{\mathrm{in}}}/_{u_{\mathrm{DC}}}$.
- The rotor leakage inductance used by FAST®, `LsigF`.
- The stator resistance used by FAST®, `RsF`.
- The measured stator resistance (if activated at start-up) `Rs_n`, which is transferred to FAST®, at completion of the estimation sequence (duration set in the `100 Hz`) module with unit `Timer-Rs`.
- The rotor resistance value used by FAST®, `RR-F`.
- The measured per unit current vector amplitude, `Ivec_n`.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 6.67 is an 'ADC-Sim' module, which is used to provide artificial inputs (as pre-set in this module) to the ADC channels. Hence these

values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes (see figure 6.68).

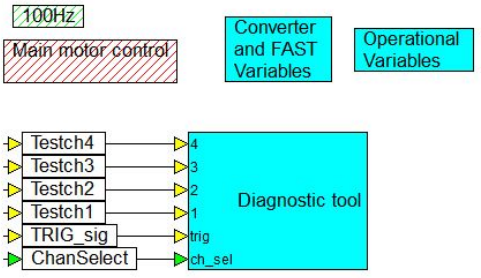


Fig. 6.68 One level down into the 'IM-FOC-FAST' Controller module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 6.69), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current(s)/frequency. In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the gain/bandwidth setting are introduced together with the variable `Res_est_current`, which defines the DC inject current amplitude used for Rs estimation at start-up (if selected).

The 'Converter and FAST Variables' dialogue box (see figure 6.70), defines the scaling used by the Analogue to Digital converter, as discussed earlier (see equation (4.2)). This dialogue box also shows the low-pass filter corner frequency `Voltage_filter (Hz)` used by the converter for measurement of the half-bridge voltages (see figure 4.12).
Furthermore, the speed controller pole `Speed filter pole (Hz)` is introduced, which is a LPF of the estimated shaft speed, set here to 50 Hz. Also added to this dialogue module are motor parameters: motor pole pair number p, rotor flux `psiR (Wb)` and rotor resistance R_R of a four parameter induction machine model. Note that the rotor flux estimate used by FAST®is tied to the reference direct axis current setting, as discussed earlier (see the 'Post Analysis' 6.2.1.4).
In the sequel of this HIL a simple approach to determining the rotor resistance R_R will be discussed. This dialogue box also sets the variable

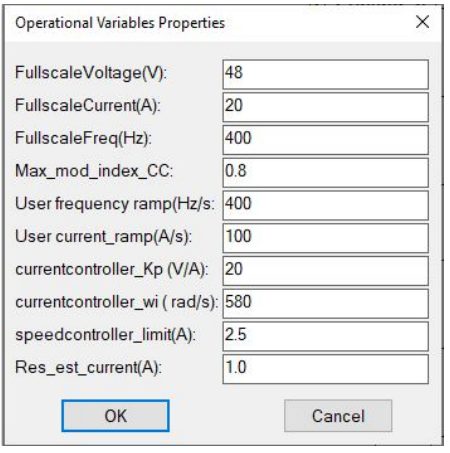


Fig. 6.69 HIL: Dialogue box 'Operational Variables'

`F_est/F_sampling`, which in effect sets the sampling frequency used by FAST®, relative to the controller sampling frequency. Typically these two are selected to be equal, i.e in this case 10 kHz. The sampling frequency for the speed controller is set by the variable `f_speed/F_sampling` and in this case is 1 kHz. The variables shown in this dialogue box are the 'primary variables' required for FAST®. However, in the `100 Hz` module a set of 'secondary variables' are defined, which do not (typically) need to be changed.

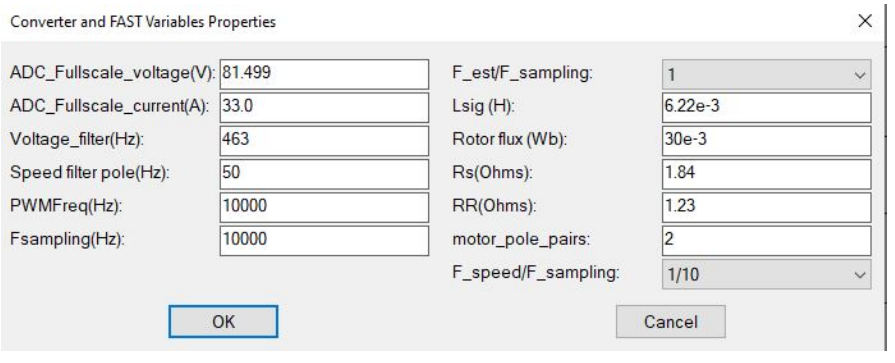


Fig. 6.70 HIL: Dialogue box 'Converter and FAST Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

The '100Hz' module, mentioned above, undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 6.68 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules (see figure 6.71). Of these, the 'FOC/speed controller IM' has been discussed earlier (see figure 6.61), where it was used in phases: PIL and COSIM of the development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module.
This module is configured with the following basic settings :

- PWM frequency: 10 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F28069` with `BOOSTXL-GANINV` converter

The converter dead-time is set to zero because gate drivers are present, which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.
The `ADC-PWM-LaunchXL-069M BOOSTXL-3phGANINV` unit is used generate the PWM signals for the converter. In addition it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively. ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the `100 Hz` module). Estimation of the ADC offset's is also controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example (see figure 6.72), and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes

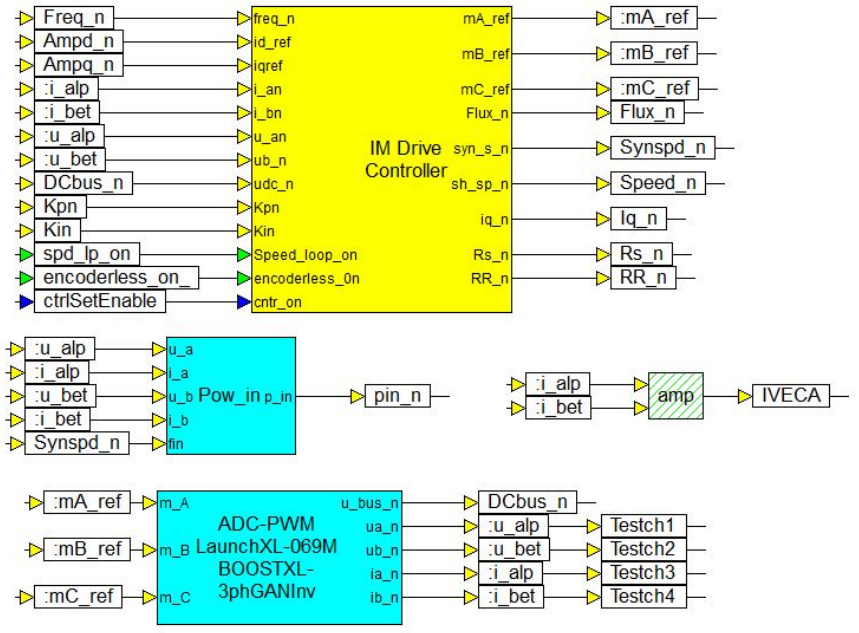


Fig. 6.71 One level down into the 'Main motor controller' module

said out-file. Three sliders are used to set the direct/quadrature axis current reference values and the shaft speed. Also shown are two sliders that set the proportional and integral gain of the speed controller. Furthermore, two sliders are added, which set the rotor resistance and leakage inductance used by FAST®.

Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

The plot given in figure 6.72, displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit LPF voltage vector locus. The shaft speed reference in figure 6.72 is set to 390 rpm.

These results shows operation under FOC-speed control with the machine operating under no-load conditions. An instructive exercise for the user is to switch operation to FOC 'torque control' (speed loop OFF) with the direct axis current set to produce rated rotor flux 32 mWb and quadrature current reference initially set to zero. Then hold the rotor by 'hand' and feel the effect of torque application when moving the quadrature reference slider.

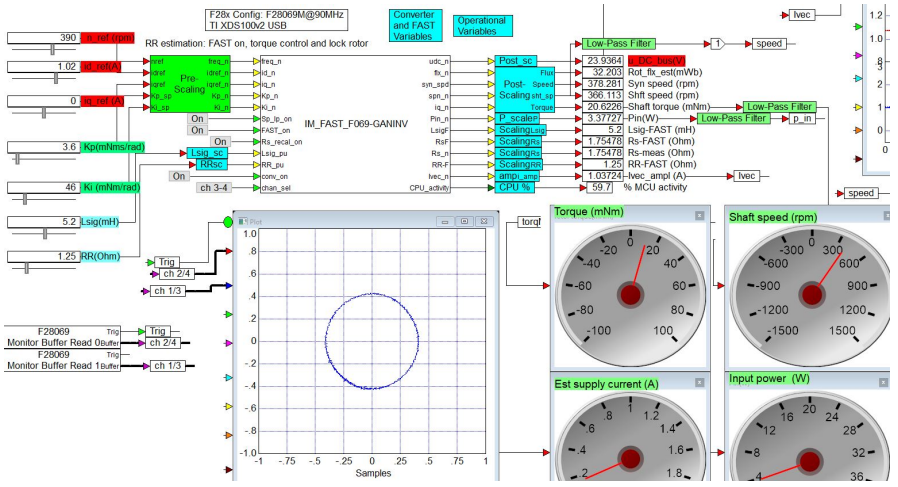


Fig. 6.72 HIL-Debug model, F069-GANINV of a induction machine drive with field-oriented control using FAST®

Post analysis using FAST®

The application of field-oriented control for induction machine drive requires knowledge of several critical parameters of the machine and this issue is raised in the 'Post analysis' section here.
An important parameter to consider is the leakage inductance give the fact that the voltage across said inductance is substantial given the substantial ratio of current to equivalent short-circuit current for this machine. This ratio is equal to $\approx 1/5$ for this machine, hence the voltage across the inductance is approximately said value times the EMF. Measurement of the leakage inductance using an L-C-R meter provides a ball park figure. Further fine tuning can be done by running the drive sensorless and under speed control, no-load and then adding a (in this case) 'hand induced mechanical load', which basically adds a quadrature current component to the current.
This effect is shown in figure 6.73, which is a plot also provided in the HIL-debug/run file (see figure 6.72) and gives the current amplitude and the rotor flux normalised to the rotor flux value set in the dialogue box (see figure 6.70). Note that this variable will change if the direct axis current reference is varied.

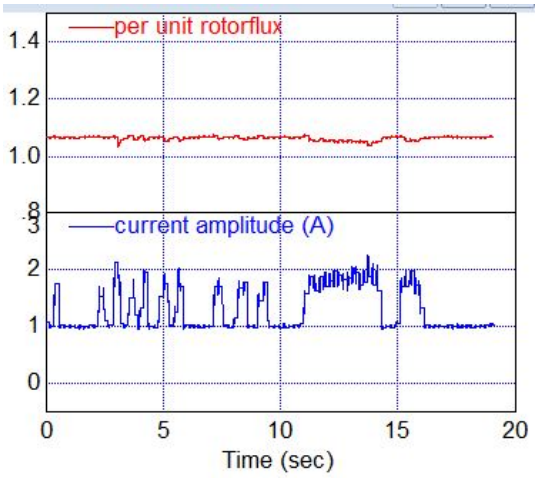


Fig. 6.73 HIL-Debug model, tuning the leakage inductance used by FAST®

Varying the current amplitude, as shown in said figure, will affect the EMF (and therefore the rotor flux amplitude) used by FAST®if the leakage inductance does not match the value experienced by the machine. Consequently, the user should incrementally change the leakage inductance slider to minimize variations of the scaled flux when machine load is applied.

The second issue to consider is the rotor resistance.
Earlier (see section 6.2.1.4), a calculation was introduced, which provides an estimate for R_R. A simpler approach to determining this value is to run the machine under sensorless torque control (speed-loop OFF) AND lock the rotor shaft. The latter requires that the estimated shaft speed MUST be zero as can, for example, be observed from the shaft speed gauge. The relationship between synchronous speed and shaft speed (see equation (6.17)), shows the importance of the rotor resistance. Hence with the rotor physically locked, the rotor resistance slider should be incrementally changed until the shaft speed gauge reads zero.

Another important issue to consider is the magnetization characteristic of the machine as introduced earlier (see figure 6.57). This characteristic can be found by operating the machine at, for example, 50% rated shaft speed, under sensorless speed control and incrementally varying the direct axis reference current from typically, 50 % rated to 150 % and recording the corresponding rotor flux values (see figure 6.57) for the machine used in this example. The magnetizing characteristic is important, as it provides the user with indication of saturation levels in the machine, relative to the choice of 'rated' direct axis current used.

Another key issue is the question of maximum speed, output power, given a set of operational settings such as: maximum modulation index, DC supply voltage and direct/quadrature currents used for the drive under consideration. To answer this question a file: `IM-max_estimator.vsm` (see figure 6.74) is provided, which makes use of the machine parameters found for the example IM under consideration.

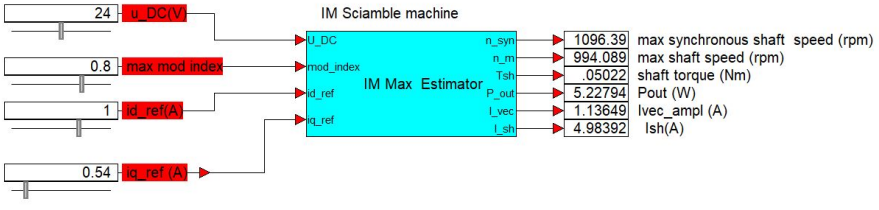


Fig. 6.74 IM Max speed Estimator

When running this file with the slider settings shown (that corresponds to what is used in the HIL, see figure 6.72), the user has an accurate indication of maximum FOC drive limits. Also shown in figure 6.74, is the equivalent short-circuit current for this IM machine.

Prior to activating this type of laboratory, the reader needs to be aware

of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

6.4.1.4 IM Sensorless FOC with FAST-F069-8301HC: HIL

Reference files:

- `IM_FAST_F069-DRV8301HC-compile.vsm`, for HIL-compile
- `IM_FAST_F069-DRV8301HC-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 6.75 represents the `DRV 8301-HC` converter (using a `LAUNCHXL-F28069` ISO card), which is connected to a 0.75 kW induction machine.
On the basis of the name plate data [5] the following information can be read or deduced:

- Star connected, 2 pole pair machine.
- Peak phase voltage/current: 24 V/35 A.
- Rated shaft torque/speed : 5 Nm/1425 rpm.
- Rated stator flux: 78 mWb.
- Rated stator frequency: 50 Hz.

Measurement using an L-C-R meter connected to two terminals and then halving the result, yields an estimate for the phase resistance R_s=55.5 mΩ and leakage inductance, L_σ=0.45 mH. These are required together with the controller sampling frequency for calculation of the current controller gain and bandwidth.

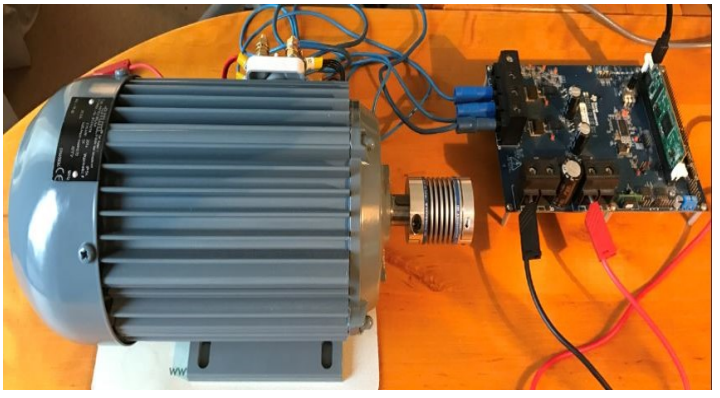


Fig. 6.75 F28069-8301HC setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches.
A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model (see figure 6.67), shows the 'Drive controller with FAST' module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct/quadrature axis reference currents `id`, `iq` and the reference shaft speed `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller. Six logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other four are used to select field-oriented control (FOC) operation with FAST® `encoderless_on`, speed control operation with FOC `Sp_lp_on` and `RS_recalb_on`, as discussed earlier (see section 6.4.1.1). The function: `Enable_Rs_actual` transfers the measured stator resistance at start-up, when ON.
Outputs of the Controller module are:

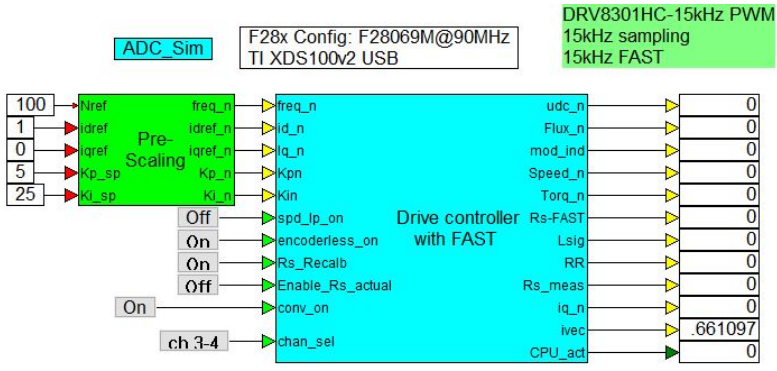


Fig. 6.76 HIL-Compile model of a DRV8301HC Induction machine drive with field-oriented control and FAST®

- The per unit DC bus voltage, `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- The per unit rotor flux estimate, `Flux_n`.
- The per unit shaft speed estimate `speed_n`.
- The modulation index amplitude, `mod_ind` used to calculate the terminal voltage amplitude.
- The estimated shaft torque, `Torq_n`.
- The stator resistance used by FAST®, `Rs-FAST`.
- The leakage inductance used by FAST®, `Lsig`.
- The rotor resistance value used by FAST®, `RR`.
- The measured stator resistance (if activated at start-up), `Rs_meas`, which is transferred to FAST®, at completion of the estimation sequence, if the function `Enable_Rs_actual` is ON.
- The measured per unit quadrature current amplitude, `iq_n`.
- The measured per unit current vector amplitude, `Ivec`.
- A `CPU_act` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 6.76 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Hence these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes (see figure 6.77).
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation.

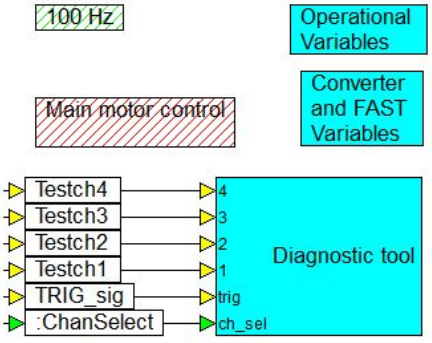


Fig. 6.77 One level down into the 'Drive controller with FAST' module

The 'Operational Variables' dialogue box (see figure 6.78), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current(s)/frequency. In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the gain/bandwidth setting are introduced together with the variable `Res_est_current`, which defines the DC injection current amplitude used for Rs estimation at start-up (if selected).

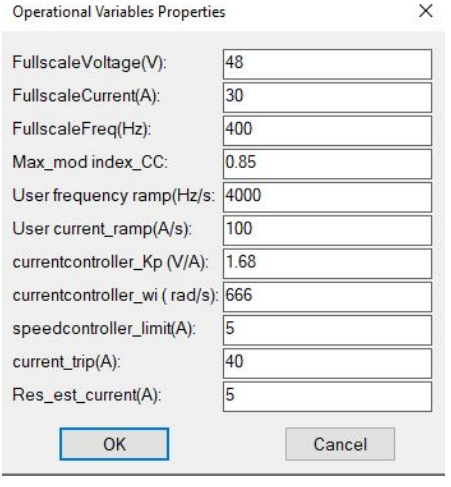


Fig. 6.78 HIL: Dialogue box 'Operational Variables'

The 'Converter and FAST Variables' dialogue boxes (see figure 6.79), defines the scaling used by the Analogue to Digital converter, as discussed earlier (see equation (4.2)).
This module also shows the low-pass filter corner frequency `Voltage_filter (Hz)`

used by the converter for measurement of the half-bridge voltages. Furthermore, the speed controller pole `Speed filter pole (Hz)` is introduced, which is a LPF of the estimated shaft speed, set here to 50 Hz. Also added to this dialogue module are motor parameters: motor pole pair number p, rotor flux `psiR (Wb)` and rotor resistance R_R of a four parameter induction machine model. Note that the rotor flux estimate used by FAST®is tied to the reference direct axis current setting, as discussed earlier (see the 'Post Analysis' 6.2.1.4). This dialogue box also sets the variable `F_est/F_sampling`, which in effect sets the sampling frequency used by FAST®, relative to the controller sampling frequency. Typically these two are selected to be equal, i.e in this case 10 kHz. The sampling frequency for the speed controller is set by the variable `f_speed/F_sampling` and in this case is 1 kHz. The variables shown in this dialogue box are the 'primary variables' required for FAST®. However, in the `100 Hz` module a set of 'secondary variables' are defined, which do not (typically) need to be changed.

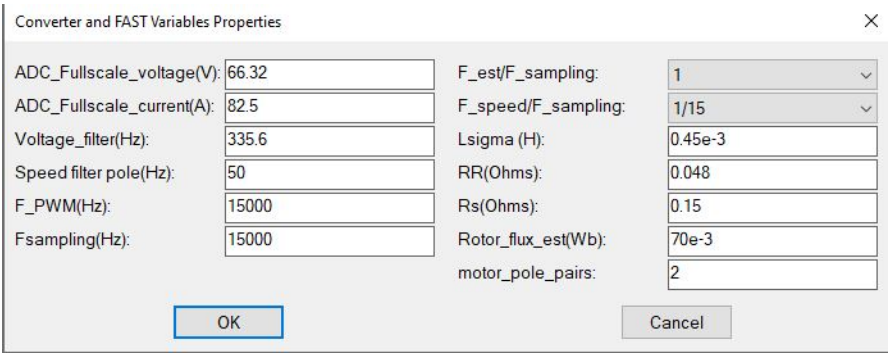


Fig. 6.79 HIL: Dialogue box 'Converter and FAST Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 6.77 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower

into the 'Main motor Control' module reveals a set of modules, shown in figure 6.80. Of these, the 'Drive Controller ' is similar to the module discussed earlier (see figure 6.61), where it was used in phases : PIL and COSIM of the development tool-chain. However, in this case the sliders used to set the rotor resistance and leakage inductance are omitted.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 15 kHz
- ADC sampling rate: 15 kHz
- Converter dead-time setting: 0.1 μs
- Board in use: `F28069` ISO control card with `DRV-8301HC` converter

This ADC-PWM module conforms with MCU timing (see figure 2.40), which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

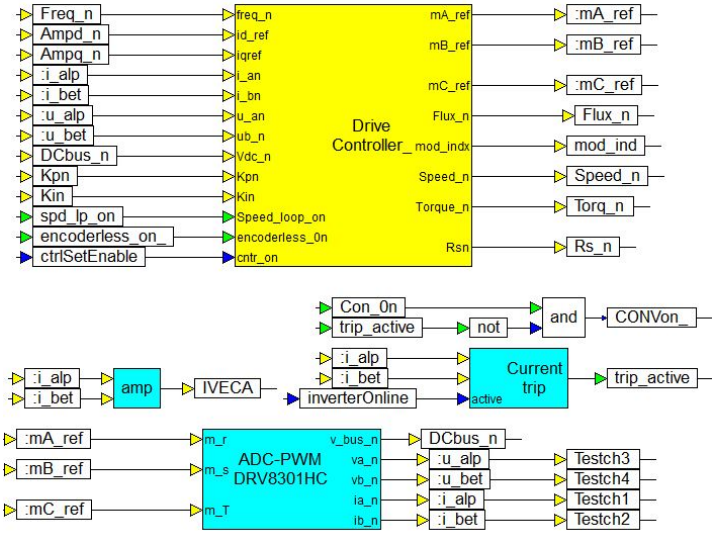


Fig. 6.80 One level down into the 'Main motor controller' module

The `ADC-PWM-DRV8301HC` unit is used generate the PWM signals for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively.

ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the `100 Hz` module). Estimation of the ADC offset's is also controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain element is the operational component of this example (see figure 6.81) and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct/quadrature axis current reference values and the shaft speed. Also shown are two sliders that set the proportional and integral gain of the speed controller.
Post-scaling modules are used to convert the per-unit measured DC bus voltage, % CPU time and other variables (as discussed in the compile section above) to actual values, as shown with numeric displays. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `chan_sel` input) diagnostic signals.

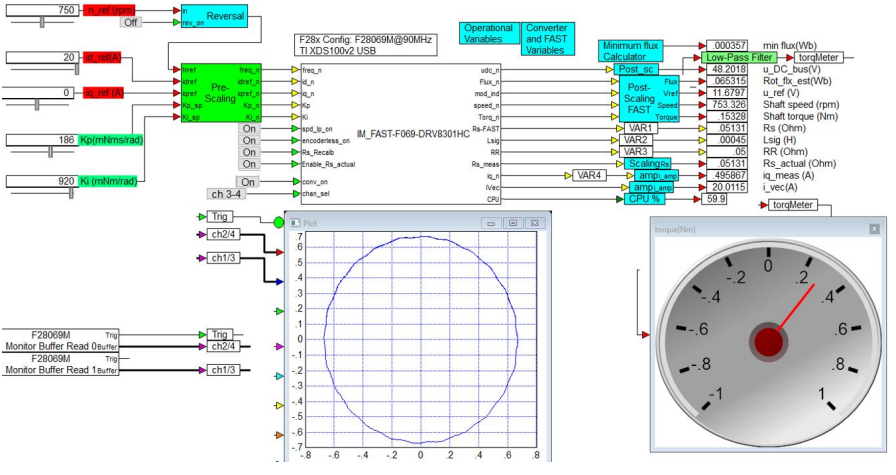


Fig. 6.81 HIL-Debug model, F069-DRV8301HC of a induction machine drive with field-oriented control using FAST®

The plot given in figure 6.81 displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit LPF voltage vector locus. The shaft speed reference in figure 6.81 is set to 750 rpm.
These results shows operation under FOC-speed control with the machine operating under no-load conditions. In addition, a speed reversal was carried out, where the quadrature current limit for the speed controller was purposely

held to 5 A (due to power supply limitations). A DC current probe with oscilloscope (vertical scale set to 10 A/div) was used to record this no-load speed reversal, as may be observed from figure 6.82.

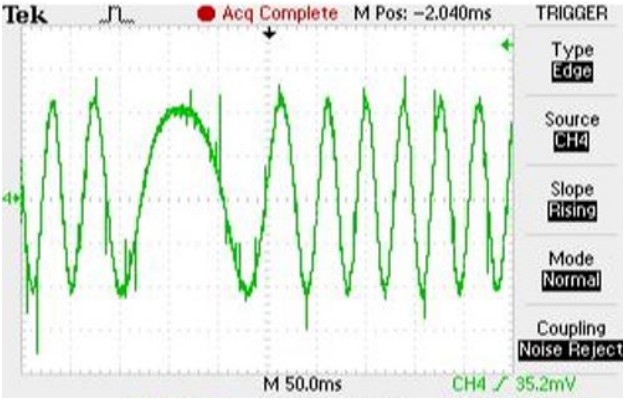


Fig. 6.82 Measured result, F069-DRV8301HC induction machine drive with field-oriented control using FAST®

Post analysis using FAST®

An important issue is the question of maximum speed, output power, given a set of operational settings such as: maximum modulation index, DC supply voltage and direct/quadrature currents used for the drive under consideration. To accommodate this question a file: `IM-max_estimator-ISEA.vsm` is provided (see figure 6.83), which makes use of the machine parameters found for the example IM under consideration.

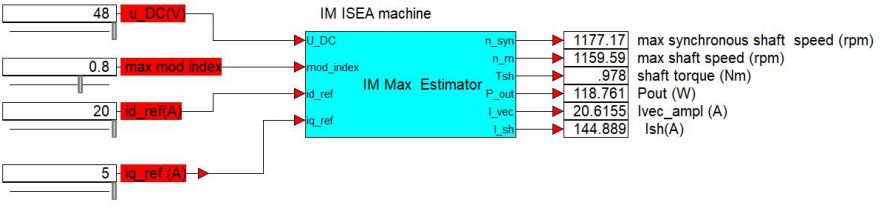


Fig. 6.83 IM Max speed Estimator

When running this file with the slider settings shown (that corresponds to what is used in the HIL, see figure 6.81), the user has an accurate indication of maximum FOC drive limits. Also shown in said figure is the equivalent short-circuit current for this IM machine.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 48 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

6.4.1.5 IM Sensorless FOC with FAST-F069-HVkit: HIL

Reference files:

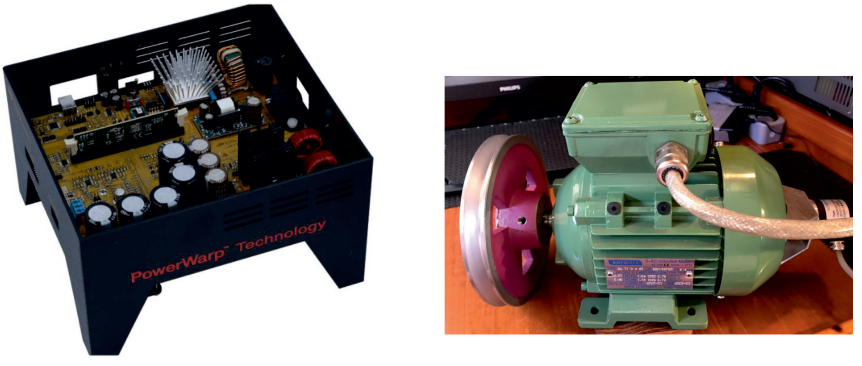
- `IM_FAST_F069_HV-compile.vsm`, for HIL-compile
- `IM_FAST_F069_HV-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) (see figure 6.84), represents the `Texas Instuments HV kit` with a `LAUNCHXL-F28069` ISO card. The converter of this `HV kit` is connected to a 0.37 kW three-phase squirrel cage induction machine [6] provided by 'Zener Electric', Sydney, Australia. Note that the motor in question is provided with an incremental shaft encoder and FOC operation with a shaft sensor is discussed in section 6.2.1.5. In this case, sensorless FOC is discussed, hence the attached encoder is not used or required.
The converter of this Texas Instruments 'High Voltage Control and PFC

Developers kit' [22] is connected to a 350 V, 2 A DC power supply.
On the basis of name plate data [6] the following information can be deduced:

- Star connected, 2 pole pair machine.
- Peak phase voltage/current: 339 V/1.4 A.
- Rated shaft torque/speed : 2.65 Nm/1330 rpm.
- Rated stator frequency: 50 Hz.

Measurement using an L-C-R meter connected to two terminals and then halving the results yields an estimate for the phase resistance R_s=7.3 Ω and leakage inductance L_σ=75 mH. These are required together with the controller sampling frequency for calculation of the current controller gain and bandwidth (see subsection 2.1.4).



(a) Texas Instruments HV kit (b) ROTOTECH Three-phase 0.37 kW IM

Fig. 6.84 MCU-converter setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches.
A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model (see figure 6.85), shows the 'FOC sensorless control' module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the per unit user direct/quadrature axis reference currents `id_n`, `iq_n` and the reference shaft frequency `freq_n`. In addition, the proportional and integral speed gain inputs `Kp_n`, `Ki_n` are provided to tune the speed controller. Furthermore, two inputs `Lsig_pu`, `RR_pu` are provided, which are used to set values (via sliders) for the leakage inductance and rotor

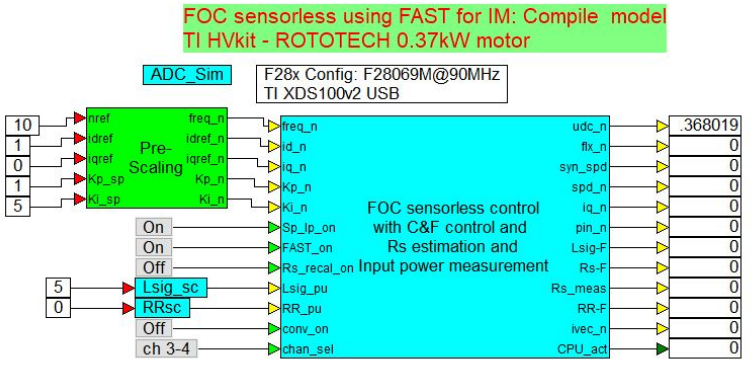


Fig. 6.85 HIL-Compile model of a HV Kit based Induction machine drive with field-oriented control and FAST®

resistance respectively.
Five logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other three are used to select field-oriented control (FOC) operation with FAST® `FAST_on`, speed control operation with FOC `Sp_lp_on` and `RS_recal_on`, as discussed earlier (see section 6.4.1.1).
Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- The per unit rotor flux estimate `Flx_n`.
- The per unit synchronous shaft speed estimate `syn_spd`.
- The per unit shaft speed estimate `spd_n`.
- The per unit measured quadrature current `iq_n` used to calculate the shaft torque.
- The measured per unit motor input power `pin_n`.
- The leakage inductance used by FAST®`Lsig-F`.
- The stator resistance used by FAST®`Rs-F`.
- The measured stator resistance (if activated at start-up) `Rs_meas`, which is transferred to FAST, at completion of the estimation sequence, if the function `Rs_recal_on` is ON.
- The rotor resistance value used by FAST®`RR-F`.
- The measured per unit current vector amplitude `ivec_n`.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 6.85 is an 'ADC-Sim' module, which is used to provide artificial inputs (as present in this module) to the ADC channels. Hence these values will appear in the controller when this file is run. This approach is

useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes (see figure 6.86).

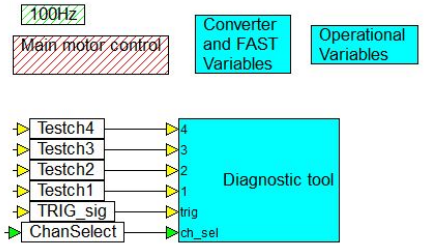


Fig. 6.86 One level down into the 'Drive controller with FAST' module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 6.87), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current(s)/frequency. In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the gain/bandwidth setting are introduced together with the variable `Res_est_current`, which defines the DC injection current amplitude used for Rs estimation at start-up (if selected).

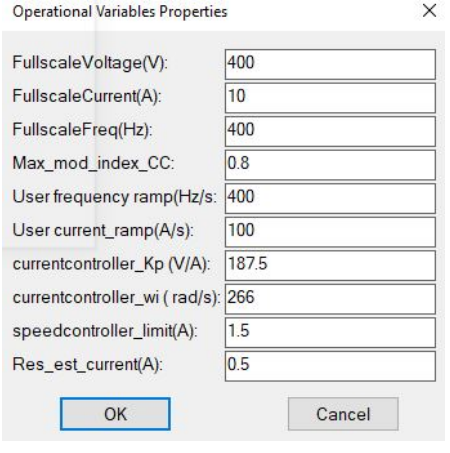


Fig. 6.87 HIL: Dialogue box 'Operational Variables'

The 'Converter and FAST Variables' dialogue box (see figure 6.88), defines the scaling used by the Analogue to Digital converter. This module also shows the low-pass filter corner frequency `Voltage_filter (Hz)` used by the converter for measurement of the half-bridge voltages.
Furthermore, the speed controller pole `Speed filter pole (Hz)` is introduced, which is a LPF of the estimated shaft speed, set here to 50 Hz. Also added to this dialogue module are motor parameters: motor pole pair number p, rotor flux `psiR (Wb)` and rotor resistance R_R of a four parameter induction machine model. Note that the leakage inductance and rotor resistance values shown in the dialogue box are 'initial' values and are replaced by the slider values when the drive becomes operational.
This module also sets the variable `F_est/F_sampling`, which in effect sets the sampling frequency used by FAST®, relative to the controller sampling frequency. Typically these two are selected to be equal, i.e in this case 10 kHz. The sampling frequency for the speed controller is set by the variable `f_speed/F_sampling` and in this case is 1 kHz. The variables shown in this dialogue box are the 'primary variables' required for FAST®. However, in the `100 Hz` module a set of 'secondary variables' are defined, which do not (typically) need to be changed.

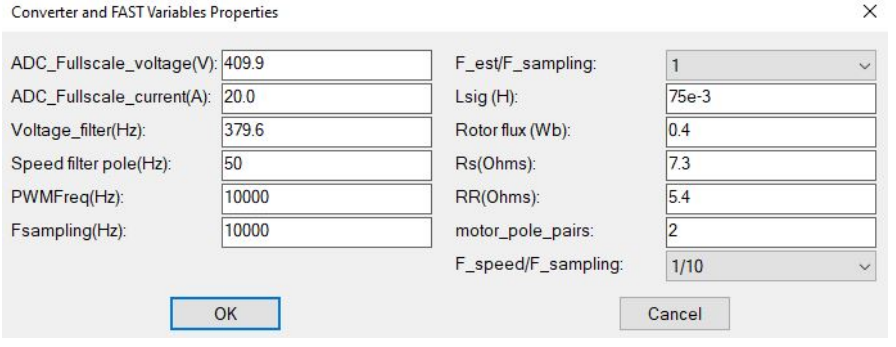


Fig. 6.88 HIL: Dialogue box 'Converter and FAST Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1`, `Test2` or `Test3`, `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

The '100Hz' module, mentioned above, undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 6.86 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 6.89. Of these, the 'Drive Controller ' is similar to the module discussed earlier, see figure 6.61 where it was used in phases : PIL and COSIM of the development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module.
This module is configured with the following basic settings :

- PWM frequency: 10 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 2 μs
- Board in use: `F28069` ISO control card with `TI HV kit` based converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

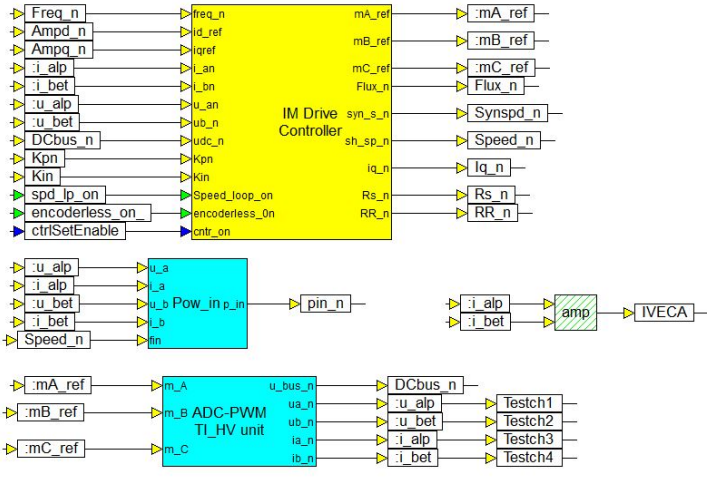


Fig. 6.89 One level down into the 'Main motor controller' module

The `ADC-PWM-TI_HV unit` is used generate the PWM signals for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet` using a forward Clarke conversion. Similarly, the three

measured (per unit) phase currents are converted to stationary coordinates α, β with outputs `i_alp`, `i_bet` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the `100 Hz` module). Estimation of the ADC offset's is also controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain element is the operational component of this example (see figure 6.90) and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes said out-file. Three sliders are used to set the direct/quadrature axis current reference values and the shaft speed. Also shown are four sliders of which two set the proportional and integral gain of the speed controller. The other two are used for fine tuning the leakage inductance and rotor resistance values, as will shortly be discussed in the 'Post analysis' alinea.
Post-scaling modules are used to convert the per-unit measured DC bus voltage, % CPU time and other variables (as discussed in the compile section) to actual values, as shown with numeric displays.
Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

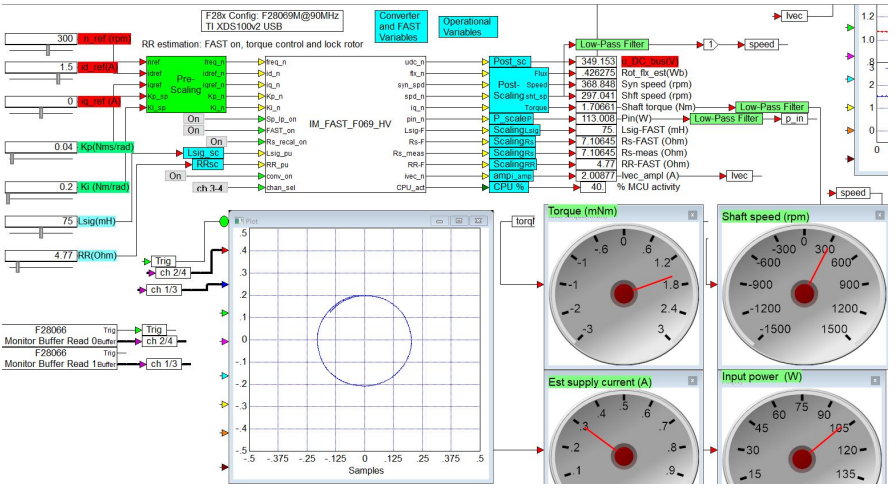


Fig. 6.90 HIL-Debug model, F069-HV kit connected to a induction machine with field-oriented control using FAST®

The plot given in figure 6.90 displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit LPF voltage vector locus. The shaft speed

reference in figure 6.90 is set to 300 rpm.
These results show operation under FOC-speed control with the machine operating under mechanical (hand)-load conditions. The presence of a mechanical load is also apparent from the speed difference between synchronous and shaft speed numeric display readings, as may be observed from figure 6.90.
In addition, a DC current probe with 'PicoScope' [20] was used to record the phase current (with applied hand-load), as may be observed from figure 6.91.

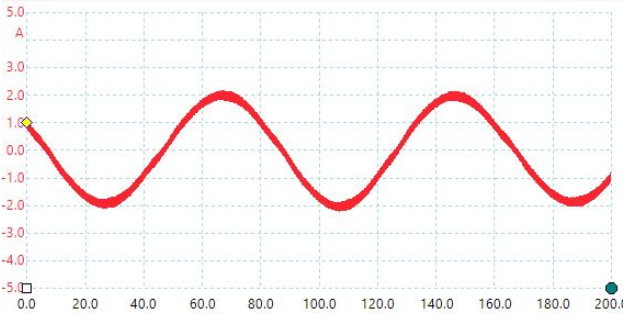


Fig. 6.91 Measured phase current using DC current probe, F28069-HV kit connected to a induction machine with field-oriented control using FAST®

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 350 V. If not

stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

Post analysis-Introduction

In this section a brief discussion is given with the purpose of identifying the rotor resistance an leakage of the machine using the HIL model shown in figure 6.90. The same HIL model will also be used to find the magnetization characteristic of the IM machine in question. In the sequel to this section an FOC drive performance estimator will be discussed.

Post analysis-Parameter estimation

Sensorless control of induction machines using FAST®or AFT®is based on the use of a four parameter induction model as shown earlier (see figure 2.24). Of these, the stator resistance R_s can be identified at start-up using the `Rs_recal_on` function. The leakage inductance L_σ is also required, to be able to estimate the EMF and an initial value of this parameter was (and should be done, as the current controller gain is calculated using this parameter) earlier by using an L-C-R meter. Fine tuning of the leakage inductance is done by running the machine sensorless under speed control and applying a (hand) load to generate a quadrature current component. If FAST®is tuned correctly (by using the slider `Lsig`), the EMF (and therefore the estimated rotor flux amplitude) should NOT change significantly. A plot (that is part of the HIL debug example) which shows the effect of load changes (see figure 6.92), confirms that the leakage slider setting of 75 mH is indeed correct.

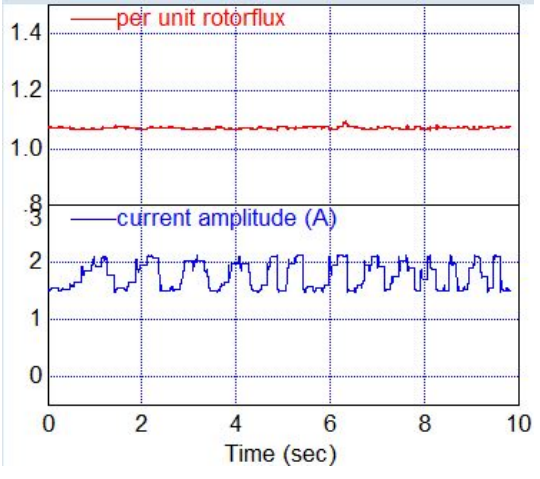


Fig. 6.92 Load current variations, introduced using a (hand) load

For shaft speed control purposes the slip frequency ω_{sl} of the drive needs to known as FAST®operation is based on the use of the synchronous frequency ω_s. Calculation of the shaft frequency ω_m is then simply, $\omega_m = \omega_s - \omega_{\text{sl}}$. The slip frequency is equal to $\omega_{\text{sl}} = {}^{i_q R_R}/_{\psi_R}$, where i_q and ψ_R represent the measured quadrature current and estimated (using FAST®) rotor flux respectively. The parameter R_R, which is the rotor resistance can be found using the slider, `RR` and operating the drive under sensorless FOC torque control AND locking the shaft. This means that the estimated shaft speed (shown on a numerical display, see figure 6.90) must be zero (because the motor shaft is locked), hence the slider `RR` should be adjusted until this condition is met. For this machine a rotor resistance of 4.77 Ω was found, using the method outlined above.

For motor modelling purposes and IM FOC sensored control, a value for the magnetizing inductance L_M is required (see figures 2.25, 6.16) and this requires identification of the magnetizing characteristic of the machine. This characteristic can be found by operating the drive under sensorless FOC, speed control and no-load conditions, typically well below rated speed, in this case a speed of 300 rpm was selected. Under these conditions the direct axis current is varied(using the slider `id_ref`, see figure 6.90) over a range that covers the rated value (which as a guide line can be the RMS value on the nameplate, in this case 1.0 A). For this drive the direct axis current was varied over the range 0.4 A→1.5 A (in steps of 0.1 A) and the estimated rotor flux was recorded for each point (using Altair Compose [18]), as may be observed from figure 6.93.
Note that such variations of the direct axis current, will require FAST®to track the corresponding rotor flux values and this implies that the dynamic range used by the algorithm should be able to handle this. A good guide for setting this range is the flux value used in the dialog box (see figure 6.88).
A linear approximation of the magnetizing characteristic has also been added and the gradient of the line represents the magnetizing inductance L_M, which for this motor was found to be 394 mH.

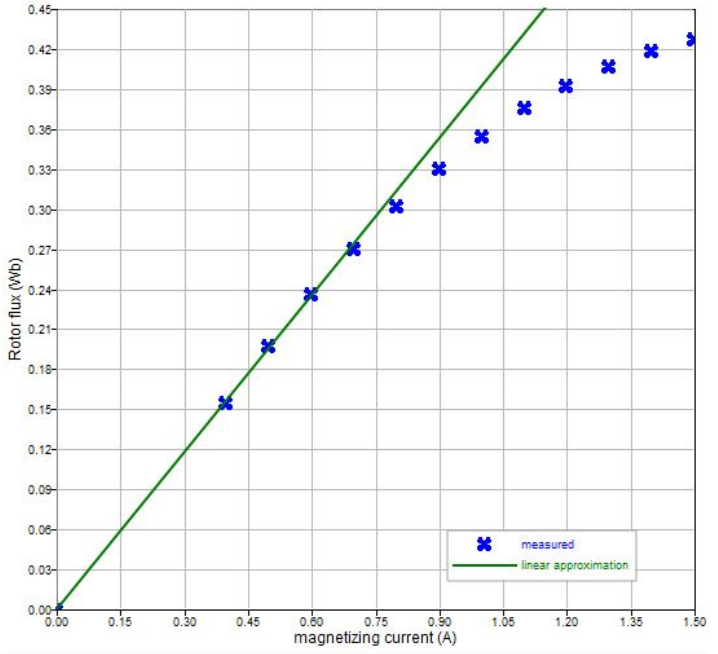


Fig. 6.93 Magnetization characteristic: measured and linear approximation, ROTOTECH 0.37kW Induction machine

Post analysis-FOC performance estimator
Reference file:

- `IM-max_estimator.vsm`, for IM FOC performance estimation.

An important operational drive issue is the question of maximum speed, output power, given a set of operational settings such as: maximum modulation index, DC supply voltage and direct/quadrature currents used for the drive under consideration. To answer this question, a file shown in figure 6.94 is provided, which makes use of the machine parameters found for the example IM under consideration. When running this file with the slider settings shown

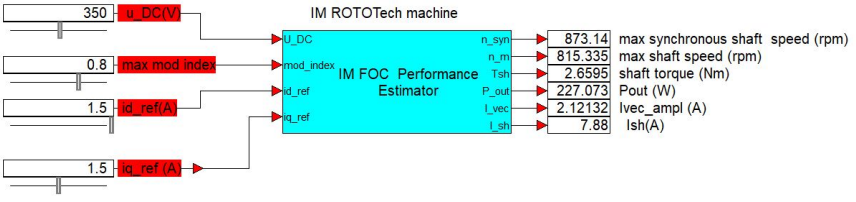


Fig. 6.94 IM-FOC performance estimator, using parameters set in HIL

(that corresponds to what is used in the HIL, see figure 6.90), the user has an accurate indication of maximum FOC drive limits, given the DC supply and converter used for this HIL example.
A salient observation from figure 6.94 is that the drive is not able to reach the rated output power of 0.37 kW (as shown on the motor name plate) and this is due to the limitation imposed by the power supply used here. If the drive is operated directly from a mains rectified DC supply, operation would be feasible under rated conditions, as may be observed from figure 6.95.

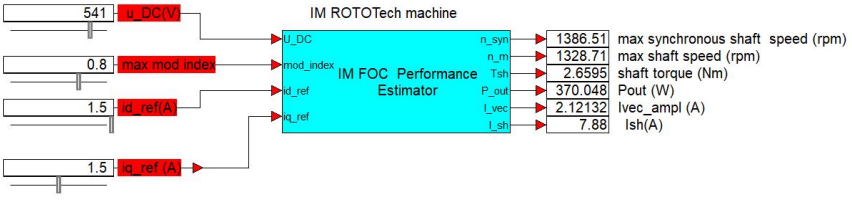


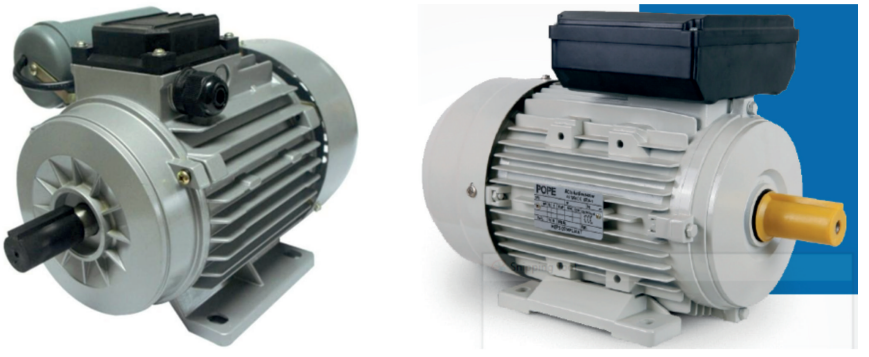
Fig. 6.95 IM FOC performance estimator, with minimum DC voltage needed to achieve rated output power

Said figure shows that the minimum DC bus voltage required (given a maximum modulation index of 0.8 , used in this drive example) to achieve rated

output power, is equal to 541 V, hence readily achievable with a 220 V/240 V mains rectified DC bus and appropriate converter.

Chapter 7
Single-phase Induction machine Drives

So called 'single-phase induction machines' (see figure 7.1), are named on the basis of utilizing a single phase mains AC supply connection.



(a) Single Capacitor IM (b) Dual Capacitor IM

Fig. 7.1 IM single phase: squirrel cage induction machine examples

However, these machines are in fact two-phase, as they make use of two windings (see figure 7.2) referred to as 'run' and 'aux' (auxiliary) respectively. Typically, these two phases are unbalanced, which implies that the winding resistance and number of turns can be different. Furthermore, these machines utilize a squirrel cage rotor, as was mentioned earlier (see figure 2.21). Machines of this type utilize either a single capacitor (see figure 7.1(a)) or two capacitors [7] (see figure 7.1(b)) to introduce a current phase shift as required to achieve a rotating magnetic field. A centrifugal switch may be also be present (as is the case for the Dual Capacitor machine) that is used to add a 'start capacitor' (parallel to the 'run' capacitor) during the start-up sequence, with the purpose of improving the starting torque of the machine, as will be discussed in the ensuing section.

In this chapter a detailed analysis on the control of single phase asynchronous (induction) machine drives will be given. Initially, operation of a 'mains connected' machine, with a quadratic load will be discussed, for the dual capacitor motor example [7] considered in this chapter. The focus of said chapter is to show operation of the 'single phase' (which implies two windings) example induction machine with a three-phase converter (see figure 7.2).

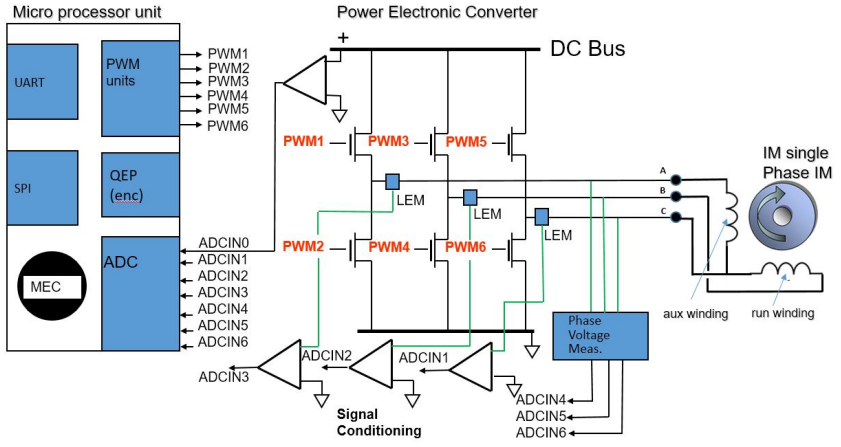


Fig. 7.2 IM single phase: three-phase converter connected wiring

The use of a two-phase machine with a three-phase converter, has been discussed earlier (see section 5) and the modulation method for single phase drives is basically the same, as shown for hybrid two-phase steppers. In terms of drive control for this induction machine, use is initially made of 'voltage & frequency control', as discussed in section 6.1 (for three-phase IM). Furthermore, sensorless field-oriented control using FAST®, as discussed in section 6.4(for three-phase IM), will be shown for the single-phase example IM machine.

7.1 Mains connected-single phase IM

Prior to discussing the use of the single-phase IM example machine [7] with a three-phase converter it is instructive to consider a simulation model of said machine when connected to a mains connected supply and using the capacitors and centrifugal switch provided. The ACTIVATE[17] based model of the single-phase example IM, connected to a 240 V RMS AC mains supply is shown in figure 7.3. A model of this single phase, squirrel cage induction machine [13], which makes use of an 'IRTF' (Ideal Rotating TransFormer) was

introduced earlier (see figure 2.28). The `IRTF_IM_Sph` model makes use of motor parameters derived from FOC sensorless operation (see subsection 7.3.1.3) with this example machine.

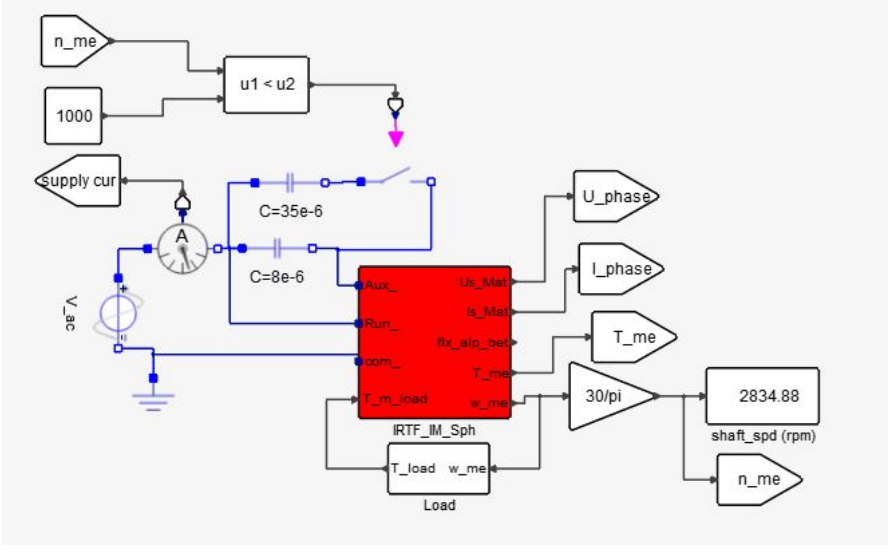


Fig. 7.3 IM single phase: mains connected example

The ACTIVATE® [17] example given in figure 7.3, shows a 'switch', that is activated by centrifugal action (see figure 7.4), which implies that the 35 uF 'start capacitor' is disconnected when a specified shaft speed is reached. In this case the 'switch open' speed value has been arbitrarily set to 1000 rpm. However, this value may vary pending the machine type used.
Hence, during the start-up transient both 'start' and 'run' capacitors are connected in parallel to improve the starting torque of the machine. In this example machine, the 'aux' winding remains connected to the 8 uF 'run capacitor' once the centrifugal switch opens.
A quadratic shaft load characteristic is present, which is set to produce a torque of 0.634 Nm at a shaft speed of 2710 rpm, which corresponds to the 0.18 kW rated output power of the example machine.
The results obtained with this ACTIVATE®based example (see figure 7.5), show the voltages/currents for both windings of the single-phase induction machine during start-up. Figure 7.5 also shows the presence of significant shaft torque ripple and a high start-up mains supply current.

The synchronous shaft speed of this one pole pair $p = 1$ machine, is in this case 3000 rpm, given the use of a 50 Hz AC mains supply. Under steady state conditions the shaft speed is at its rated value of 2710 rpm, as expected.

Fig. 7.4 Single phase dual capacitor IM : centrifugal switch assembly

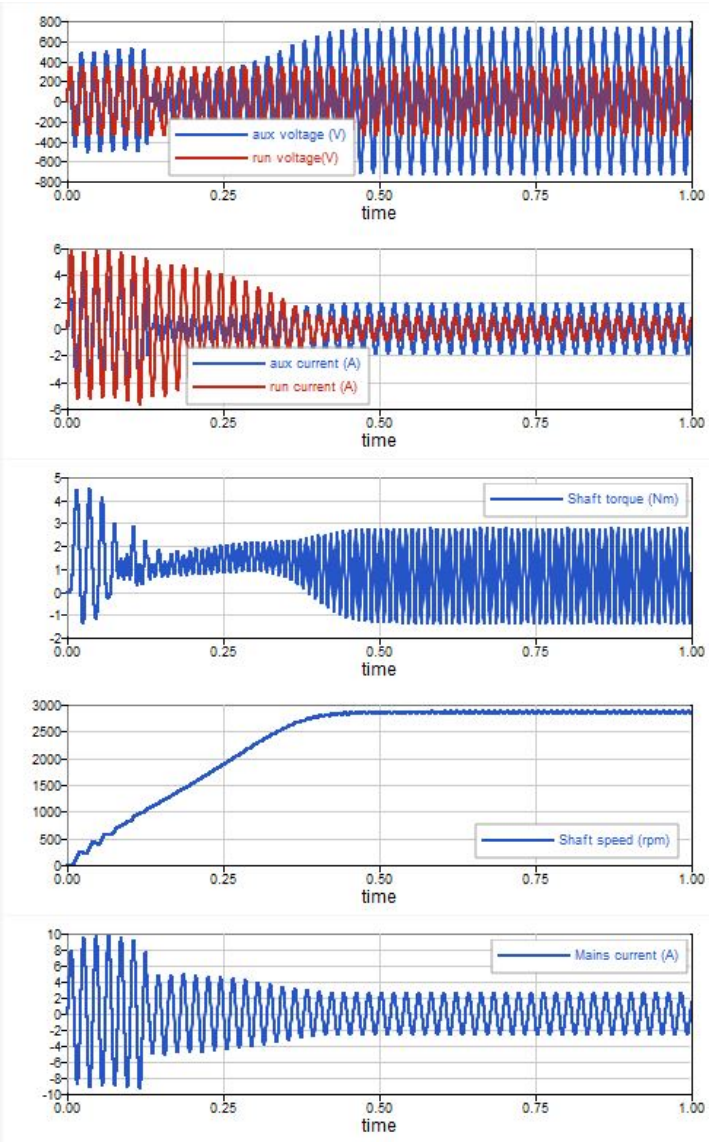


Fig. 7.5 Single phase dual capacitor IM: start-up operation with load

7.2 Voltage-Frequency Control-single phase IM

Purpose of these examples is to realize voltage/frequency control of a single phase induction motor. Use is made on the voltage control approach discussed earlier(see section 6) for three-phase machines. A modified version of this control approach is shown in this section, which is applicable for single phase voltage/frequency control [13].
A V/f controller essentially maintains the ratio of voltage and frequency (in rad/s) at an approximately constant value, given that said ratio represents the flux in the machine. The reference for the controller is rotor flux ψ_R^*, as this largely controls the EMF component, together with the electrical frequency ω_s. In terms of calculating the required d, q voltage vector components, the voltage across the leakage inductance, defined as $\vec{u}_L = j^{e\,\vec{i}_s}/_{i_{\rm sc}}$, where the equivalent short-circuit current $i_{\rm sc} = {}^{\psi_R}/_{L_\sigma}$ is ignored, given that this the term $({}^{e\,\vec{i}_s}/_{i_{\rm sh}})$ is relatively small.
When (approximately) maintaining a constant flux the shaft torque/speed characteristic $T_m\,(\omega_m)$ is as shown in figure 7.6 together with load torque speed curve $T_l\,(\omega_m)$, as introduced earlier for three-phase machines.

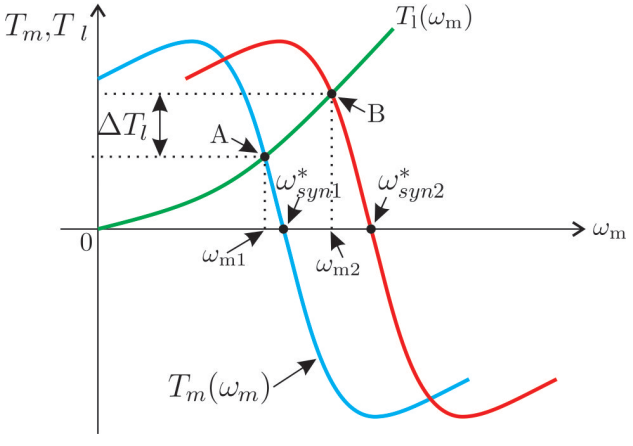


Fig. 7.6 V/f drive operation for induction machine, with reference flux $\psi_R^{\rm ref}$,

Voltage-frequency Control of the drive is done by controlling the reference synchronous speed $\omega^*_{\rm syn}$.
When controlling the reference synchronous speed (no speed sensor required) the machine torque/speed curve shifts from operating point A→B and a load/shaft speed change ΔT_l will occur (see figure 7.6). Observe that the steady state shaft speed will be different from the synchronous speed and in this case changes from $\omega_{\rm m1} \rightarrow \omega_{\rm m2}$ (see figure 7.6), where it is noted that the slip frequency $\omega_{\rm slip} = (\omega_{\rm syn} - \omega_m)$ increases with load torque.
The type of voltage-frequency controller for implementing this approach is

shown below (see figure 7.7).

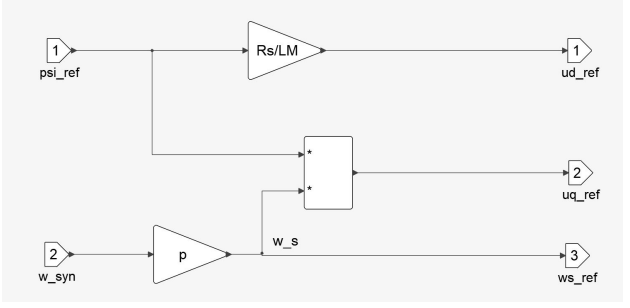


Fig. 7.7 simplified V/F induction machine control, without speed sensor

The controller representation satisfies the following set of equations [13];

$$u_d^{\text{ref}} = \psi_R^{\text{ref}} \frac{R_s}{L_M} \tag{7.1a}$$

$$u_q^{\text{ref}} = \omega_s^{\text{ref}} \psi_R^{\text{ref}} \tag{7.1b}$$

$$\omega_s^{\text{ref}} = p\, \omega_{\text{syn}}^{\text{ref}} \tag{7.1c}$$

where L_M, represents the magnetizing inductance of the (four-parameter) machine (as discussed in section 2.1.3.5) and ψ_R^{ref} represents the rotor flux reference variable (Wb). The 'stator resistance' R_s is for single phase machines replaced by the average resistances of the 'run' and 'aux' windings hence, $R_s = {}^{(R_{\text{run}}+R_{\text{aux}})}/_2$. The input synchronous reference shaft speed $\omega_{\text{syn}}^{\text{ref}}$ is multiplied by the pole pair number p, as required to generate the electrical frequency ω_s^{ref}.

7.2.1 Voltage-Frequency Control-single phase IM-TI

Purpose of this section is to realize Voltage-frequency control of a single- phase induction machine In this case the toolchain: SIM→COSIM→HIL, will be considered for a specific hardware example. For this purpose the `HVkit` with `F28069M` ISO card, is connected to a single phase Induction [7] machine. This machine has been rewired to allow direct access to both phases, which are then

connected to the three-phase converter as shown in figure 7.2. Furthermore, the non-rotating part of the centrifugal switch assembly shown in figure 7.4 has been removed (which gives a substantial reduction in friction torque) together with the two capacitors.

7.2.1.1 Voltage-Frequency Control-F069-HVkit: SIM

Reference file: `Vf_sph_F069_HVkit-SIM.vsm`

Prior to operating the single phase V-f drive, real-time, it is prudent to test the fixed point controller under simulation conditions (see figure 7.8).

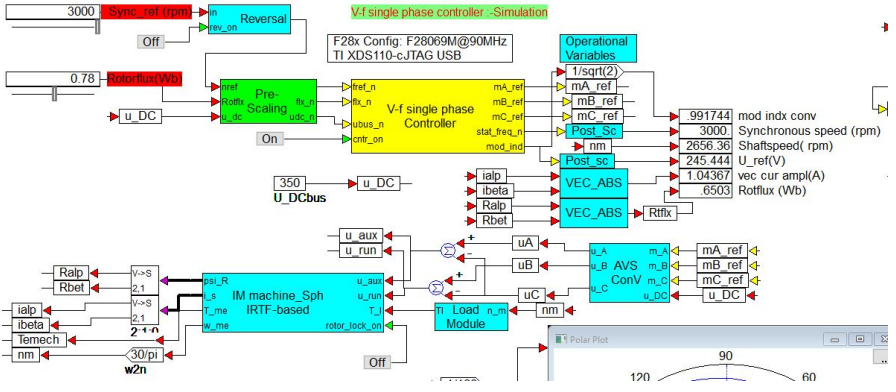


Fig. 7.8 Simulation of a single phase IM based V-f drive

A 'Pre-scaling' module is used to convert and scale the floating point, user (slider) variables to fixed point format. Inputs to the V/f controller module are:

- `fref_n`: per unit reference frequency, which is linked to the reference shaft speed $n_{\text{me}}(\text{rpm})$ by a gain factor $\left(\frac{p\, n_{\text{me}}^{\text{ref}}}{60\, f_{\text{fs}}}\right)$.
- `flx_n`: per unit rotor flux, which is scaled by the full scale flux ψ_{fs} set in the user dialog box (to be discussed).
- `ubus_n`: per unit bus voltage, scaled by the full scale voltage u_{fs} set in the user dialogue box (to be discussed).
- `cntr_on`: control active, which enables the controller.

Outputs of the V-f module are the per unit stator frequency `stat_freq_n` together with the phase modulation indices: `mA_ref`, `mB_ref`, `mC_ref`, which are connected to the 'AVS ConV' ,module. This multiplies the modulation indices by half the DC bus voltage and limits the outputs to $\pm 175V$ prior to

use by the single phase induction motor model. The two `SUM` units are used to generate the aux/run voltages for the single phase IM given the wiring arrangement shown in figure 7.2. For convenience the modulation index of the three-phase converter named `mod indx conv` is also shown on a numerical display as an absolute value of ONE is the limit achievable. For two-phase machines the modulation index of the controller (and therefore two-phase machine) is a factor $\sqrt{2}$ higher due to the presence of a two-phase pulse-centering unit.
Outputs of the single phase IRTF IM model are the currents i_α, i_β (which are the 'aux' and 'run' currents respectively), shaft torque T_{me}, shaft speed n_{me} and rotor flux $\psi_{R\alpha}$, $\psi_{R\beta}$.
The dialogue box 'Operational Variables' (see figure 7.9), contain all the user settings required for this example.

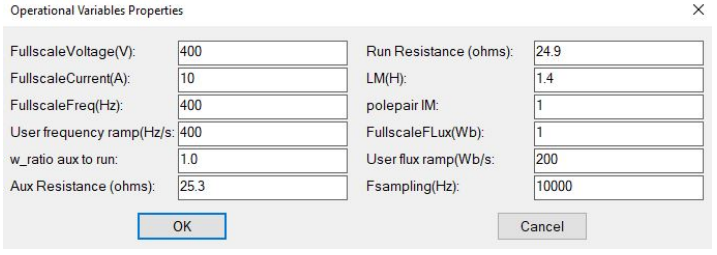


Fig. 7.9 Dialogue box used for drive with a V-f Controller

The dialogue box is also used to assign full scale values for the voltage, current, frequency and rotor flux. Dialogue box entries for the maximum allowable rate of change for flux and frequency are also provided. Said box also assigns the motor pole pair number and parameters of the single phase example induction motor. In addition, the sampling frequency used for the V/f controller module, as set by the user is shown in this dialogue box.
Moving one level lower into the 'V-f single phase Controller' module reveals a set of modules (see figure 7.10).
The reference synchronous speed signal is connected to the `freq_n` input of the frequency rate limiter module 'F-RAMP' and switch (with input `con_act2`), which ensures that said variable is held at zero when the converter is disabled. The `Park-rev` module transforms the d, q modulation indices from the `Vf-estimator` to α, β coordinates. The α modulation index, that is used for the 'aux' winding is scaled by a factor `wr_motor`, which represents the auxiliary/run winding ratio. A two phase pulse centering model as introduced earlier (see figure 5.5), generates the three phase converter modulation indices.
The basic structure of the `Vf-estimator` (see figure 7.11), reflects the control structure of the floating point version (see figure 7.7) where the outputs are given in terms of voltages, not modulation indices, as is the case for the fixed point module.

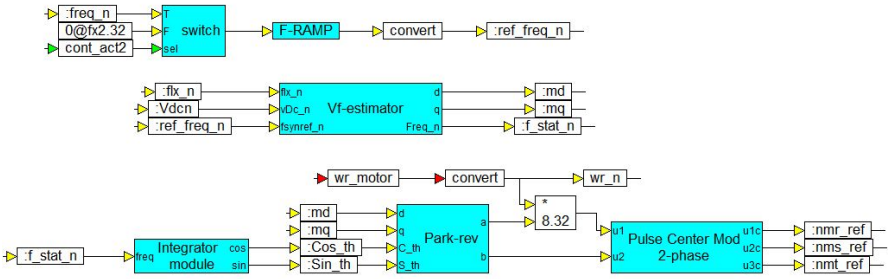


Fig. 7.10 One level into the 'V-f single phase Controller'

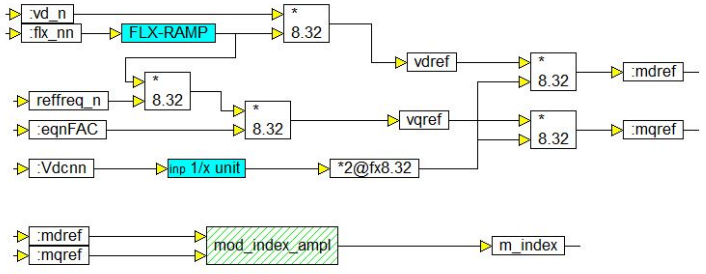


Fig. 7.11 One level into the 'Vf estimator'

The per unit reference synchronous speed signal and rotor flux inputs are represented by variables `reffreq_n_n` and `flx_nn` respectively.
A brief outline of the variables which appear in the discrete estimator are given below:

- `vd_n`= $\frac{R_s}{L_M}\left(\frac{\psi_{\text{fs}}}{u_{\text{fs}}}\right)$, where ψ_{fs}, u_{fs} represent the full scale flux and voltage values respectively. For single phase machines the variable R_s is replaced by the average of the 'run and 'aux' winding resistances.
- `flxnn`= $\frac{\psi_s^{\text{ref}}}{\psi_{\text{fs}}}$, which is the per unit rated scaled rotor flux reference value.
- `reffreq_n`= $\frac{f_s^{\text{ref}}}{f_{\text{fs}}}$, which is the per unit reference stator (synchronous) frequency. This variable is used as an input to the 'integrator' module of the voltage controller.
- `egnFAC`= $\frac{2\pi\psi_{\text{fs}}}{u_{\text{fs}}}$, which is a scaling factor used to scale the term $e_q = \omega_s\,\psi_s^{\text{ref}}$.
- `Vdcnn`= $\frac{u_{\text{DC}}}{u_{\text{fs}}}$, which is the per unit measured DC bus voltage generated by the ADC/PWM module.

A set of 'Rate Limiter' modules, entitled 'F-RAMP' (sets the frequency ramp rate) and 'FLX-RAMP' (sets the flux ramp, with input the per unit user flux value `flx_n`, have been introduced in figures 7.10, 7.11 to avoid erratic

changes to the user input variables (as actually used by the controller), which may damage the converter or motor. The rate of variable change is specified in the 'Operations Variables' dialogue box (see figure 7.9).
Outputs of the V/f module are the modulation index variables `md_ref` ,`mq_ref`. The third output is the per unit stator reference frequency `f_stat_n`, which is connected to an integrator that generates the Cos/Sin angles of the reference voltage vector (see figure 7.11). Also shown in said figure is a variable `m_index`, which is the modulation index vector amplitude used to calculate the reference voltage vector amplitude.

An example of drive operation is shown in figure 7.12, for an operation mode with the largest linear modulation index (which is 1.4 for two-phase drives). The quadratic load/speed characteristic is set to produce 0.634 Nm at 2710 rpm. The sampling time T_s as defined in the program pull-down menu is set to 100 μs, i.e a 10 kHz ADC sampling frequency is used for the discrete controller model. For this example the flux reference is set to 0.78 Wb and observation of the time plot shows that there is a discrepancy between reference and observed flux levels. This is due to the fact that the voltage across the leakage inductance is not taken into account for this controller. An important observation is that the current amplitudes with V-f control are much lower when compared with the main connected results (see figure 7.5). Furthermore, torque pulsation as seen in the main-connected case, are removed by making use of a three-phase converter with a single-phase IM. Note that the DC bus voltage was set to 350 V, given that this is the value used for HIL operation.
A vector plot which corresponds to the results given in figure 7.13 at time mark 1.0 s shows the current vector 'blue', rotor flux vector 'green' and the terminal voltage vector 'red'(scaled by a factor $1/100$).

An interesting observation of the voltage vector amplitude is that its value is 245.0 V, which is (as was mentioned earlier) at the limit of linear three-phase operation and this corresponds to a modulation index value of 1.4 (exact value is $\sqrt{2}$), which is the theoretical limit when using a two phase modulator. This index value is higher then the value that can be used for three-phase AC drives, which is $2/\sqrt{3}$, hence operation at higher speed levels is possible when using a two-phase machine with a three-phase converter. Note that operation with high modulation values, will cause current measurement problems when using shunts in the lower switches of the converter (see subsection 2.1.5 that outlines this issue).

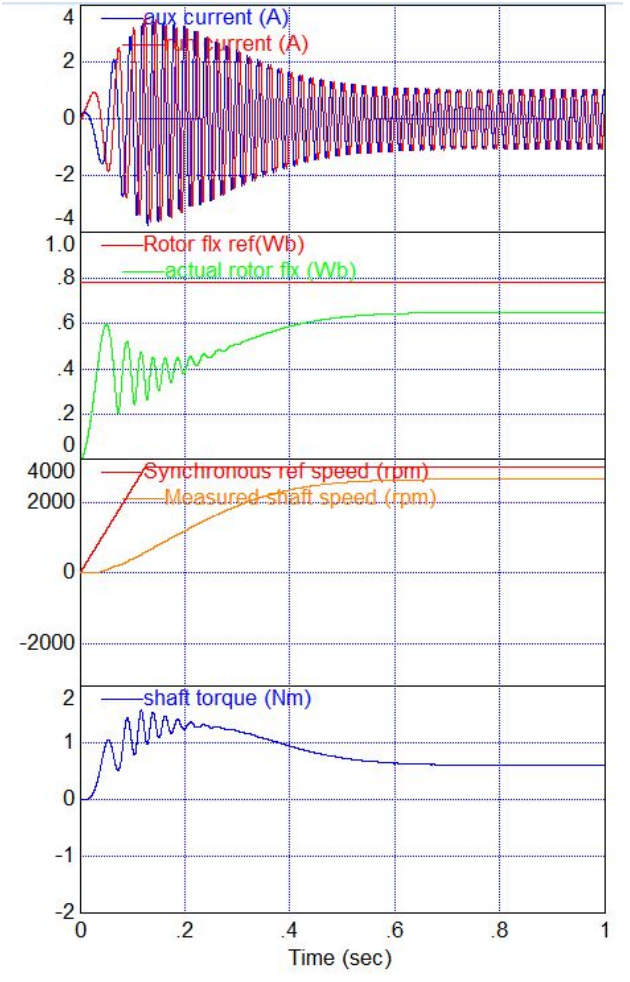


Fig. 7.12 Simulation: Typical waveforms obtained using a V-f controller

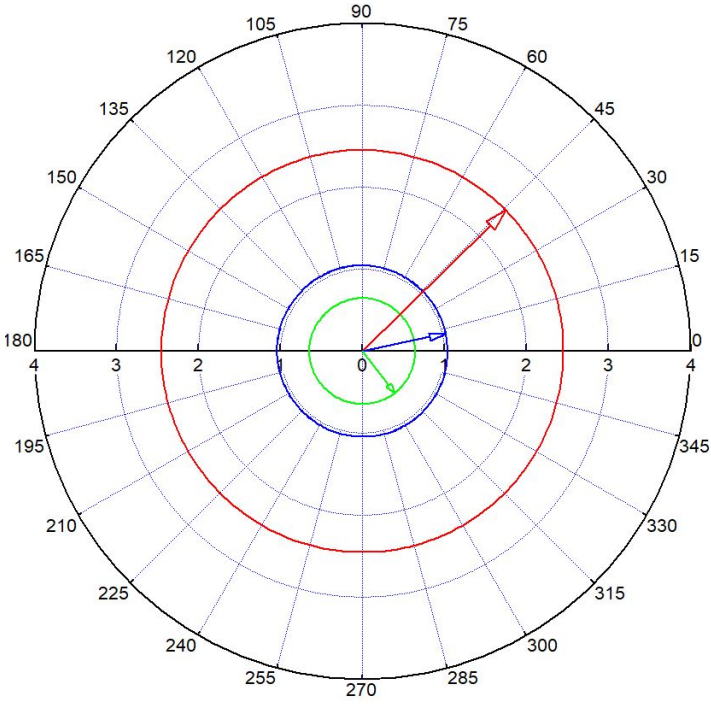


Fig. 7.13 Simulation: Vector plot showing V-f controller operation

7.2.1.2 Voltage-Frequency Control-F069-HVkit: COSIM

Reference files:

- `V&F_Sph-069_HVkit-COSIM.vsm`
- `HVkit_INVv1.psimsch`
- `my motor model_Sph.psimsch`

To evaluate operation of a controller with a switching converter, in this case the Texas Instruments `HVkit`, use can be made of a `PSIMCoupler` module, as indicated in figure 7.14. This approach implies the use of PSIM together with Embed, which is referred to as 'co-simulation'.

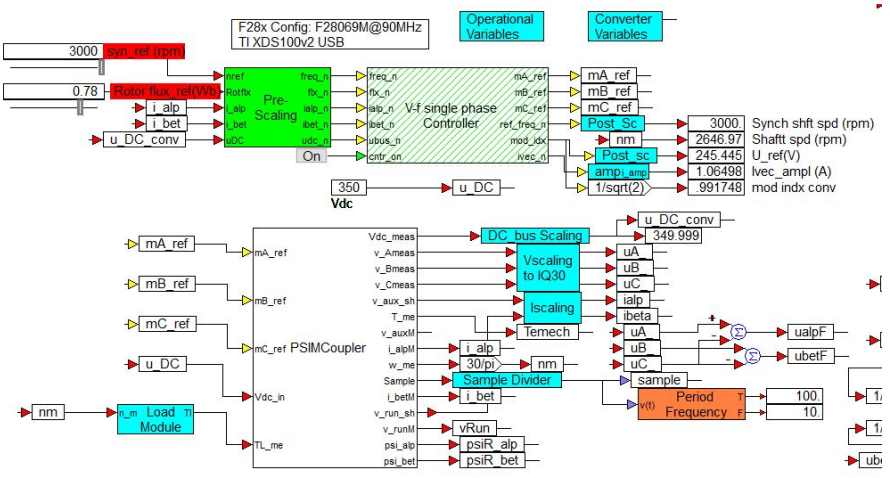


Fig. 7.14 COSIM based single phase IM drive, Voltage-Frequency control

The `V-f single phase controller`, hatched in this case as its sampling frequency is locally set to 10 kHz, whereas the 'system' frequency (set in the 'system properties' dialogue box) is set to 20 MHz. The latter is much higher, given the need to accurately represent 10 kHz voltage switching waveforms generated by the PWM unit of the converter. The content of the `V-f single phase controller` module was introduced earlier (see figure 7.10). However, the current and voltage measurements now make use of the signal processing circuitry connected to the half-bridges.
In this case shunts in the lower converter switches are used to measure currents, as shown in figure 7.15. After sampling and signal-processing in the converter, these variables appear as: `ialp`, `ibeta` in figure 7.14 and represent the measured 'aux' and 'run' currents respectively. The low pass filtered voltages (which removes the PWM component) of the three half-bridges is

also present in figure 7.14 and are also scaled and sampled by the module `Vscaling to IQ30`. In this case the per unit variables are rescaled to actual values: `uA_`, `uB_` and `uC_` respectively. Note that the LPF 'aux' and 'run'voltage are derived from these three converter voltages using `SUM` units and are represented by variables `ualpF` and `ubetF` respectively (see figure 7.14).
The 'Converter Variables' dialogue box (see figure 7.14), defines the scaling used by the Analog to Digital converter, which has a maximum input value of 3.3 V and is directly linked to the signal processing undertaken for voltages and currents as discussed above. For example, the 'ADC-fullscale-voltage' value is the theoretical peak-to peak input voltage which must be applied to the low-pass filter network, to ensure that the ADC input level varies between $0 \rightarrow 3.3$V.
The current shunt voltage(s) `v_aux_sh`, `v_run_sh` (see figure 7.15) are sampled by the `Iscaling` module shown in figure 7.14. When dealing with the actual MCU, the ADC converter also takes over this sampling process. Apart from the measured voltage/currents generated by the converter, the PSIM coupler unit also provides the actual 'aux', 'run' motor model currents (in stationary coordinates), 'run' voltage and speed, which are presented by the variables `vRun`, `i_alp`, `i_bet` and `n_m` (after conversion to rpm) respectively, see figure 7.14.

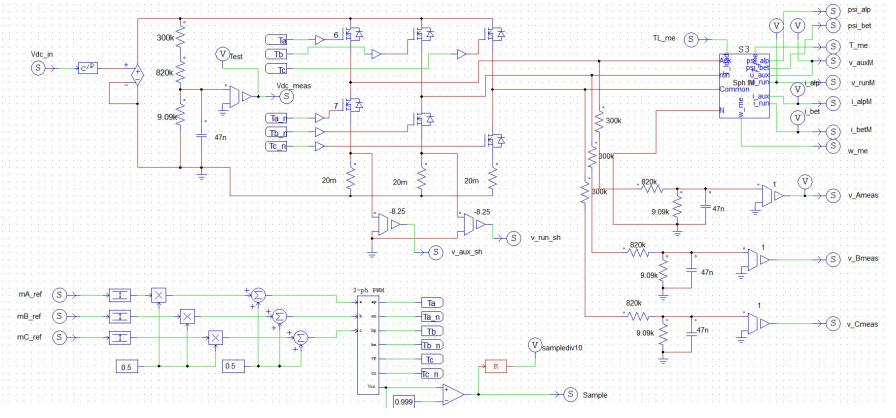


Fig. 7.15 Content PSIM model of HVkit with single phase induction machine

The PSIM induction machine model `Sph IM` used in the coupler module, as shown in figure 7.16 is based on the IRTF model discussed earlier (see figure 2.28).
Results achieved with the COSIM approach are shown in figure 7.17.
Indicated in said figure is the 'run' winding voltage waveform together with the reference voltage, 'run' winding current, shaft torque and shaft speed.
A vector plot, given in figure 7.18, at time mark 0.7 s after starting the co-

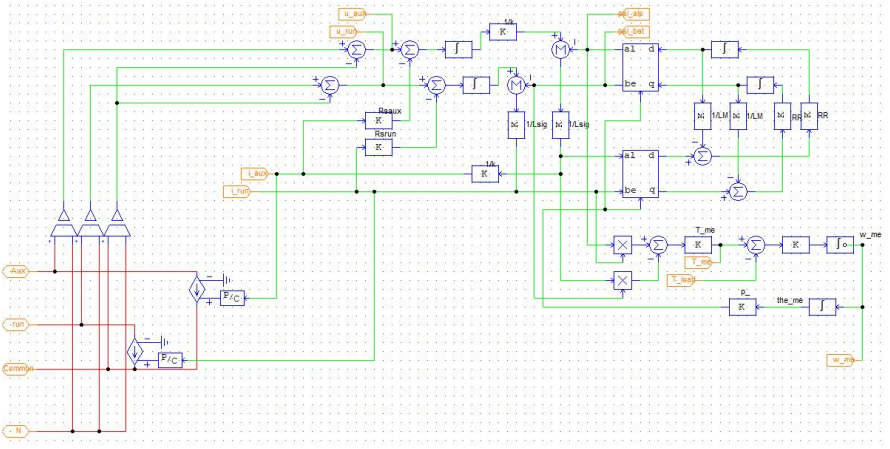


Fig. 7.16 PSIM model IRTF based single phase squirrel-cage induction machine

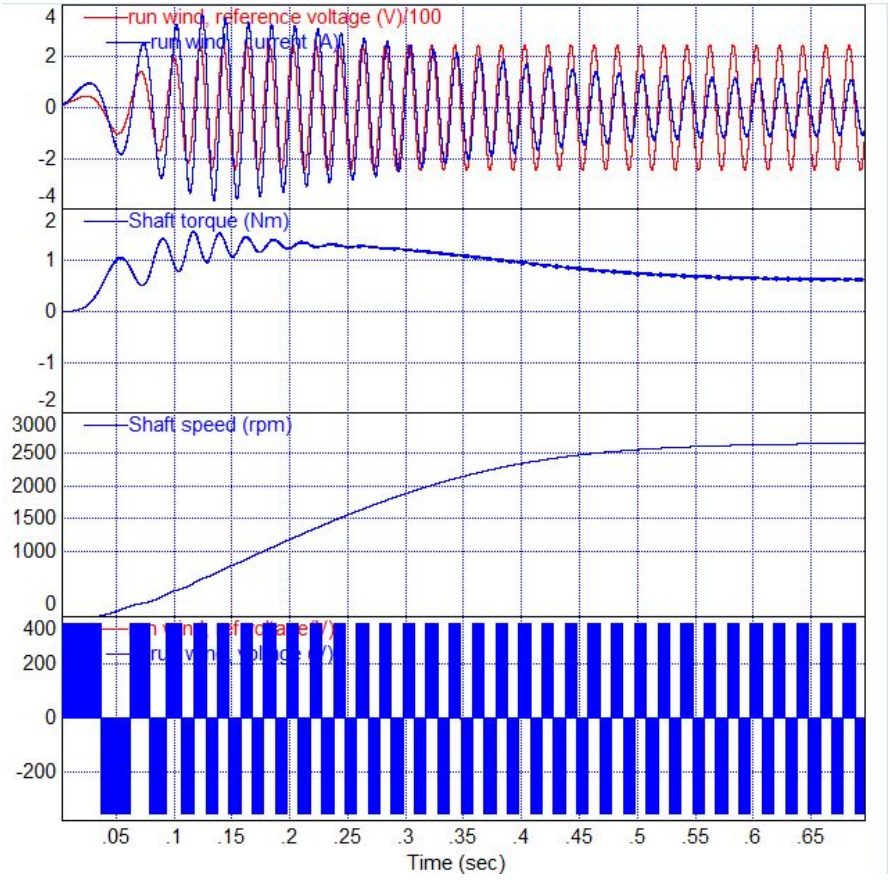


Fig. 7.17 result COSIM, single phase IM Voltage-Frequency control

simulation shows the measured LPF voltage vector ('blue' and scaled by a factor 1/100), rotor flux vector ('green' and scaled by a factor 2) and the actual (from the IRTF motor model) current vector ('red').

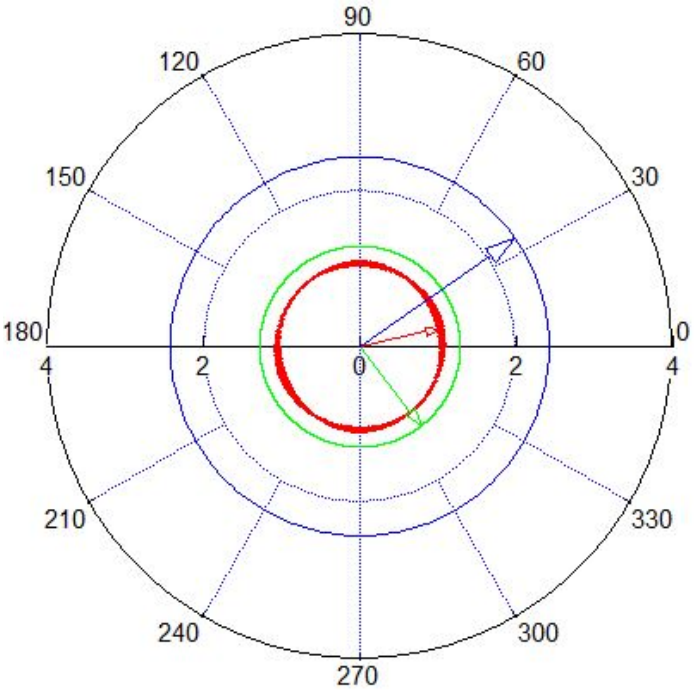


Fig. 7.18 result COSIM, single phase IM Voltage-Frequency control: vector plot

Co-simulation, as discussed in this sub-section, is instructive for understanding the impact of using a switching converter on drive operation,e.g to asses the impact of shaft torque ripple due to PWM.

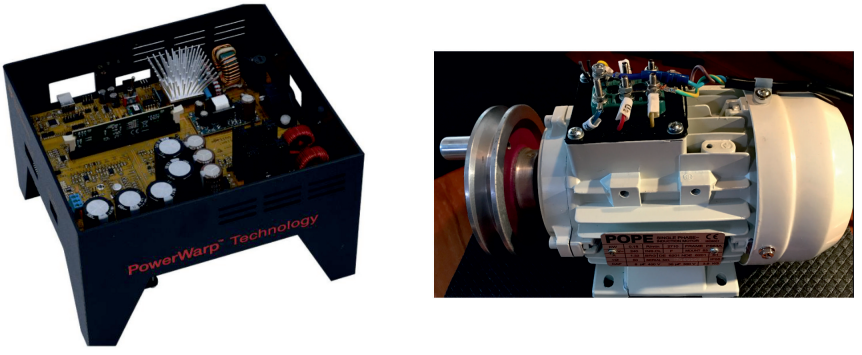
7.2.1.3 Voltage-Frequency Control-F069-HVkit:HIL

Reference files:

- `VandF_Sph__HV_kit_069M-compile.vsm`, for HIL-compile
- `VandF_Sph__HV_kit_069M-run_d.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 7.19 represents the Texas Instruments `HV kit` converter (with `F28069M` ISO card), which is electrically connected to a single phase Induction machine [7]. The machine in question has in fact been mentioned earlier (see

figure 7.1(b)), but has been rewired for converter operation, as shown in figure 7.19(b). Furthermore, the stationary part of the centrifugal switch (see figure 7.4) and the two capacitors have been removed as they are not required, when using a three-phase converter as discussed in this subsection.



(a) Texas Instruments HV kit (b) POPE Single -phase 0.18 kW IM

Fig. 7.19 MCU-converter setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. electrical wiring details are important to ensure that the direction of rotation is in accordance with controller requirements. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 7.20 shows the 'Open loop voltage-freq. controller' module, which must be compiled to generate an out-file .

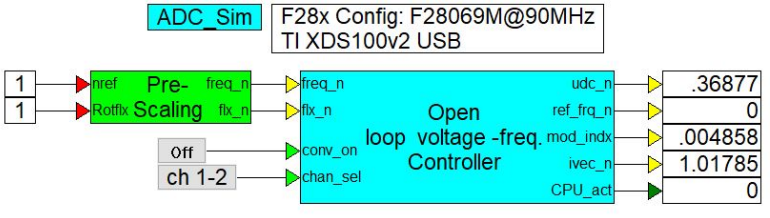


Fig. 7.20 HIL-Compile model of a single phase induction machine drive with Voltage-Frequency control

Inputs, shown in the stage as constants (for readability purposes) for this

controller module are the user synchronous reference speed `freq_n` and the reference rotor flx `flx_n`, which have been converted to 'per unit' values in the 'pre-scaling' module. Two logic inputs are provided, which allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed.
Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- The per unit synchronous speed `ref_frq_n` generated by the V-f controller.
- The controller modulation index `mod_indx`.
- The per unit measured current vector amplitude `ivec_n`.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 7.20 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes (see figure 7.21).

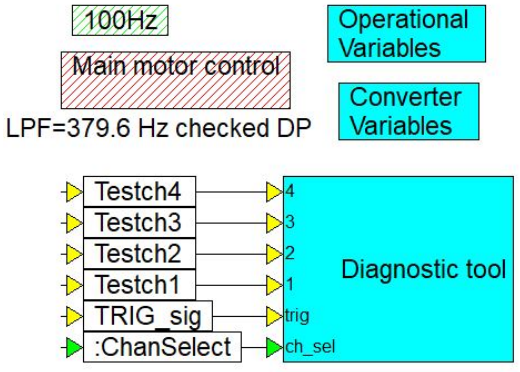


Fig. 7.21 One level down into the FOC sensored Controller module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box, mentioned earlier (see figure 7.9), is used to assign full scale values

for the voltage, current, frequency and the ramp change for the reference voltage/frequency. The 'Converter Variables' dialogue box (see figure 7.22), defines the scaling used by the Analogue to Digital converter.

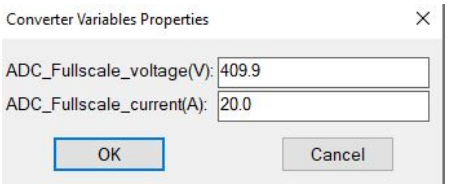


Fig. 7.22 Dialog box:'Converter variables'

A 'Diagnostic tool' module is used to buffer two variables, which can be subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 7.21 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit.
Moving one level lower into the 'Main motor Control' module reveals a set of modules (see figure 7.23). Of these, the 'V-f controller' has been discussed earlier (see figure 7.11), where it was used in phases: SIM, COSIM of the development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module.
This module is configured with the following basic settings :

- PWM frequency: 10 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 2 μs
- Board in use: `F28069 ISO` control card with `TI HV kit` based converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

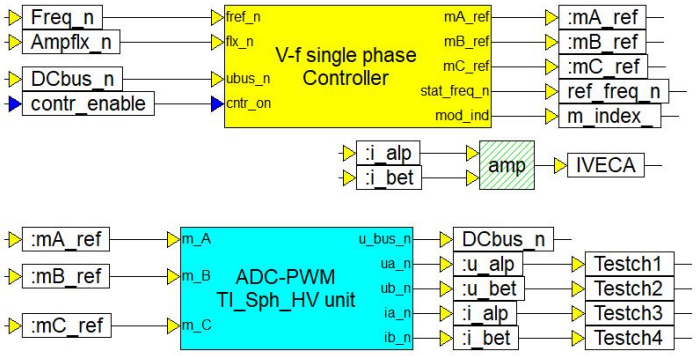


Fig. 7.23 One level down into the 'Main motor controller' module

The `ADC-PWM-TI_Sph HV unit` unit is used generate the PWM signals for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to the 'aux' and 'run' winding voltages: `u_alp`, `u_bet`. The two measured (per unit) 'aux' and 'run' currents are represented by outputs `i_alp`, `i_bet` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example (see figure 7.24), and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module', which executes said out-file. Two sliders are used to set the rotor flux reference value and the synchronous shaft speed. Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. In addition the synchronous shaft speed, reference voltage vector amplitude and measured current vector amplitude are shown on numeric displays. For convenience the modulation index `mod indx conv` of the three-phase converter has been added, and its value should not exceed ONE.
Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.
The plot given in figure 7.24, displays the per unit measured voltage vector locus. Multiplication of the locus amplitude (using the 'read coordinates' feature of the plot gives the `U_ref(V)` numeric display reading. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus. The speed and rotor flux references in figure 7.24

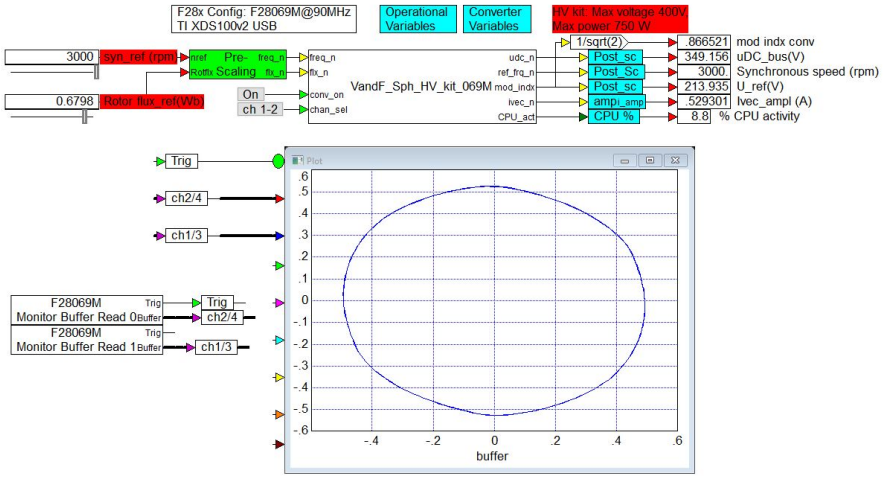


Fig. 7.24 HIL-Debug model, F28069M-HV kit of a single phase induction machine

are set to 3000 rpm and 0.679 Wb respectively. With this operation condition the three-phase converter index, `mod indx conv` is 0.866 , hence care should be taken to not significantly exceed this value, as current measurement is implemented with shunts in the lower converter switches (see figure 7.15). Note also, that the modulation index used by the motor and controller is a factor $\sqrt{2}$ higher, hence 1.22 given the use of a two phase machine with a three-phase converter.

These results show operation with the machine operating under no-load.

In addition, a DC current probe with 'PicoScope' [20] was used to record the 'aux' current (under no-load) using 20 kHz LPF active filter (a feature of the Picoscope), as may be observed from figure 7.25.

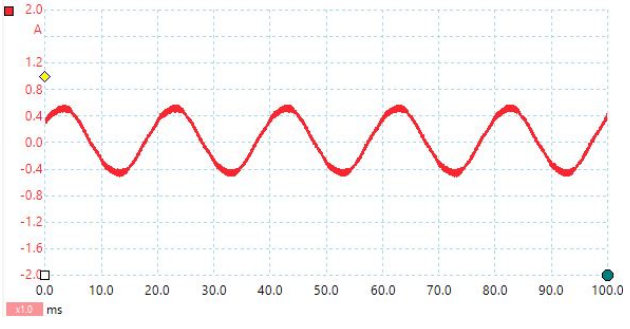


Fig. 7.25 Measured auxiliary ('aux') current using DC current probe, F069-HV kit connected to a single phase induction machine with Voltage-Frequency control

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 350 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

7.3 Sensorless Field Oriented Control single phase IM

The term 'sensorless' or more appropriate 'encoderless' control specifically refers to the exclusion of a shaft speed or encoder sensor normally used to achieve field-oriented control of a IM drive. Encoderless FOC control is principally based on identification of the EMF vector $\vec{e}$ in terms of its amplitude and orientation in the α, β coordinate system, as discussed earlier for three phase IM drives, see section 6.4. Once the EMF vector has been identified the rotor flux is estimated by integrating said EMF vector, using:

$$\vec{\psi}_R^{\text{est}} = \frac{\vec{e}}{j\omega_s} \tag{7.2}$$

Sensorless IM field-oriented control is then established, whereby the direct axis current component (which is tied to the estimated rotor flux vector, hence in alignment) determines the magnitude of said flux vector.

For single phase induction machines, EMF estimation is more elaborate because there are two windings present, referred to as the 'auxiliary' (aux) and 'run' winding, as mentioned earlier (see figure 7.2). The resistances of these windings may differ and furthermore, the number of turns may also be different. Consequently, EMF calculation for single phase IM requires evaluation of expression (7.3).

$$\vec{e} = \vec{e}_R - \vec{u}_L \tag{7.3a}$$

$$\vec{e}_R = \frac{1}{k}\left(u_{\text{aux}} - i_{\text{aux}} R_{\text{aux}}\right) + j\left(u_{\text{run}} - i_{\text{run}} R_{\text{run}}\right) \tag{7.3b}$$

$$\vec{u}_L = j\omega_s L_\sigma \left(k\, i_{\text{aux}} + j\, i_{\text{run}}\right) \tag{7.3c}$$

where R_{aux}, R_{run} represent the winding resistances of the 'aux' and 'run' winding respectively. Furthermore, the variable k represent the 'aux'/'run' winding ratio of the machine. Measurement of the converter voltages and aux/run currents (in stationary α, β coordinates, as shown in figure 7.2) and knowledge of the leakage inductance L_σ allows estimation of the EMF vector using expression (7.3).
Equation (7.2) shows an important issue with respect to rotor flux estimation. When the electrical frequency moves from $\omega_s \to 0$ the EMF amplitude will likewise move: $|\vec{e}| \to 0$. Which implies that expression (7.2) becomes undefined at zero synchronous frequency, which for an induction machine typically differs from the shaft speed frequency (which depends on the applied mechanical load). Hence, an induction machine with load can, for example, operate under locked rotor conditions, because the synchronous speed will then be unequal to zero.
Sensorless control for IM makes use of the 'synchronous' d, q reference frame and the shaft speed of the machine is calculated using expression (7.4).

$$\omega_{\text{me}} = \left(\omega_s - \frac{i_q\, R_R}{\psi_R}\right)\frac{1}{p} \tag{7.4}$$

where ω_{me} is the shaft speed in rad/s. This expression requires access to the

rotor resistance value R_R, synchronous speed ω_s (calculated by the encoderless algorithm in use in the 'HIL' sub-section) and the measured quadrature current i_q. Vice-versa equation (7.4) can be used to estimate this parameter R_R for the case where the rotor is purposely locked (which means $\omega_{\text{me}} = 0$), as will be discussed at a later stage. Expression (7.4) requires access to the rotor flux magnitude ψ_R and ω_s, which are generated by the encoderless algorithm used.
For IM FOC drives the flux is controlled by the direct axis current i_d^{ref} an example of this relationship, known as the magnetizing curve, is given in figure 7.26.

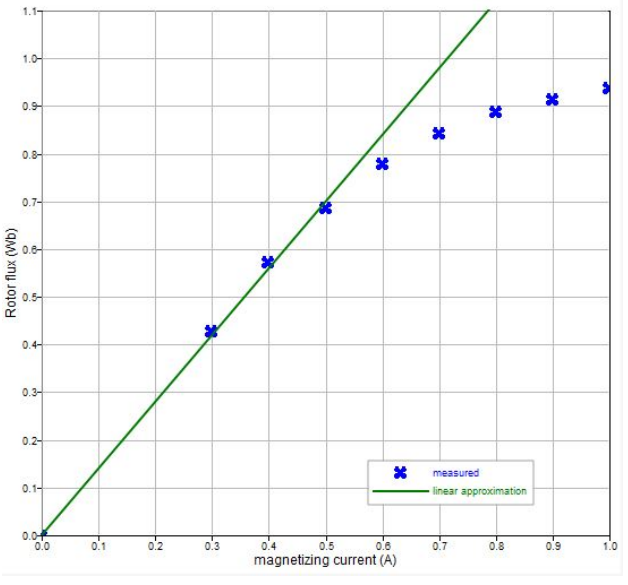


Fig. 7.26 Magnetizing characteristic of the example single phase induction machine [7]

Added to figure 7.26 is a linear function, of which the gradient represents the magnetizing inductance L_M of the machine. This parameter is required for the SIM and PIL examples, which use a linear IRTF model and for control algorithms which require rotor flux calculation using $\psi_R = L_M\, i_d$ as shown earlier, see figure 7.7. However, for encoderless operation this linear approximation is NOT required, as the rotor flux is directly estimated using the AFT®or FAST®algorithm.

7.3.1 Sensorless Field Oriented Control of single phase IM using FAST®-TI

The examples linked to Texas Instruments will be covered using the development tool-chain components: PIL→COSIM→HIL, based on the use of the Piccolo [23] F28069M processor. This MCU has access to the Texas Instruments software InstaSPIN®, which contains their classified sensorless algorithm FAST®. Access to FAST is by way of an Embed DLL, which implies that simulation is not an option when using this algorithm. Initially the tool-chain components presented in the ensuing subsections will be based on the use of the `HV kit` (with `F28069M` ISO card) drive set-up connected to a single phase induction machine [7] as shown in figure 7.2.

7.3.1.1 IM Sph FAST FOC-F069-HVkit:PIL

Reference files:

- `P-COM_IM_Sph_PIL_Launch.vsm`
- `P-COM-IM_FAST-PIL_mod.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F28069M, which contains the DLL used to communicate with FAST. This implies the use a target execution module for file: `P-COM-IM-FOC_FAST-PIL.out` as indicated in figure 7.27. The target interface module that executes the out-file, generates the modulation indices required for the converter/motor and these components are not affected by use of the PIL. Generation of the out file is based on the PIL module, see figure 7.28, which must be compiled using the command: `toolbox>codegen`.
Moving one level into the PIL module `IM-Sph-FAST_PIL` shows a 'Sph Drive Controller', see figure 7.29.
Added to said controller is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that the PIL control is operational. Furthermore, the modules responsible for input power measurement `Pow_in` and current vector amplitude `amp` are located outside the 'Sph Drive Controller'.

The content of the 'Sph Drive Controller', as given in figure 7.30 shows a `Single phase Current controller`, which is very similar to the one introduced earlier for two-phase hybrid steppers (see figure 5.22). The per unit measured current inputs are the variables `ia_n`, `ib_n`, which represent the 'aux' and 'run' current respectively. The speed controller concept `Speed Controller` has also been introduced earlier for three-phase machines (see section 4.3.1.1). Furthermore, a `Rs estimation model` is present, which for IM single phase is used to consecutively measure the 'aux' and 'run'

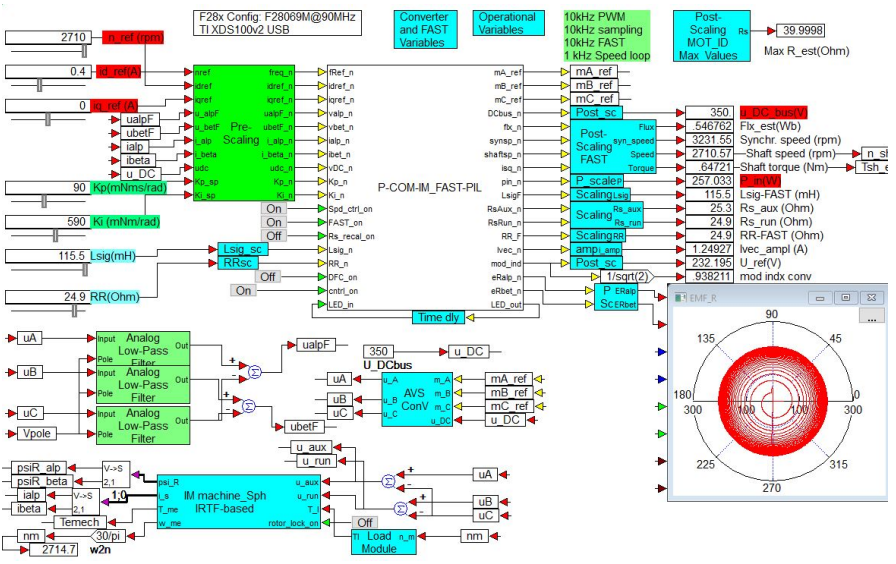


Fig. 7.27 Field-oriented torque/speed control for single phase IM using FAST®:PIL operation

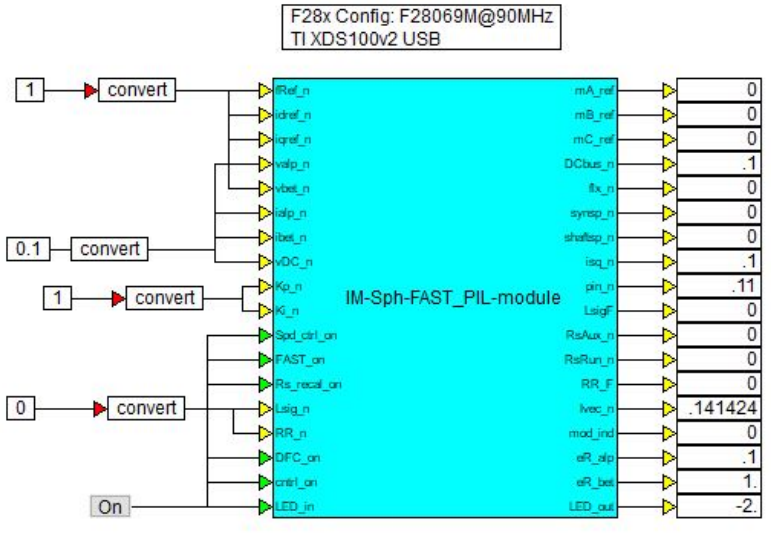


Fig. 7.28 Module:'IM-Sph-FOC-FAST-PIL'

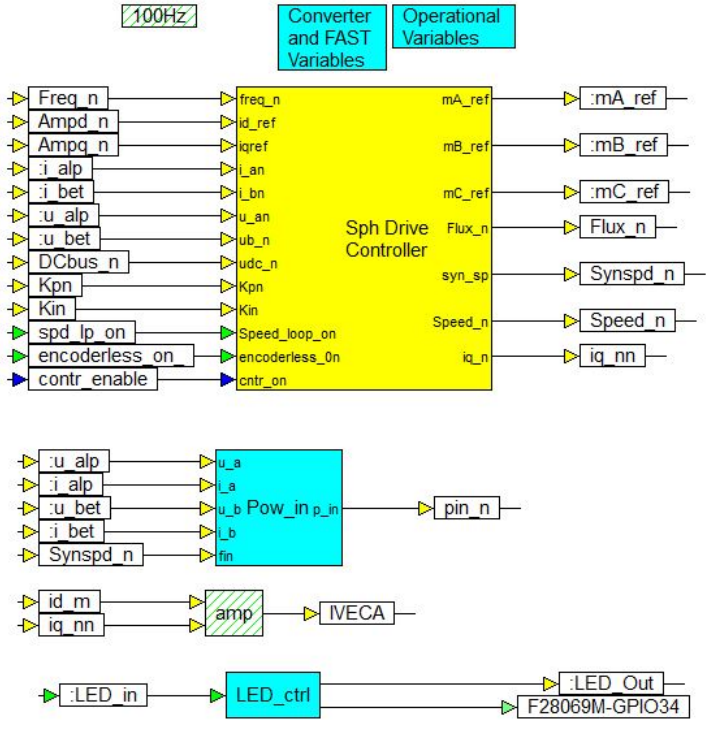


Fig. 7.29 Content PIL module, Field-oriented torque/speed control with FAST®

winding resistances at start-up (if the `Rs_recal_on` option is enabled) using DC current injection. Also present in figure 7.30 is a module: `C-Ramp_EPL` which allows 'dynamic field control' (DFC), as discussed earlier for three-phase induction machines (see section 6.3) if the option `DFC_on` is activated (see figure 7.27). The modulation index vector amplitude of the controller is calculated using the module `mod_index`.
Said figure also shows a polar plot `EMF_R`, that shows the EMF $\vec{e}_R$, as defined by equation (7.3b), which is used by the Embed FAST®DLL. Consequently, the stator resistance value in said DLL must be set to zero. The polar plot is of use for tuning the 'aux/run'winding ratio k and this is achieved by choosing a value that ensures a concentric EMF locus, as shown in figure 7.27.
The FAST®module `InstaSPIN FOC` provides per unit estimates of the rotor flux amplitude `ROTORFLX`, rotor flux angle `ang`, electrical shaft frequency `rotfreq` and synchronous frequency `fe_n`. Inputs to the FAST®module are the per unit measured LPF 'aux'/'run' winding voltages (shown as `van`, `vbn`) and 'aux'/'run' currents (shown as `ia_n`, `ib_n` in figure 7.30). In addition, a variable `CurCTLstate` (generated in the `100 Hz` module) activates the FAST®module, which is also provided with the per unit measured DC bus

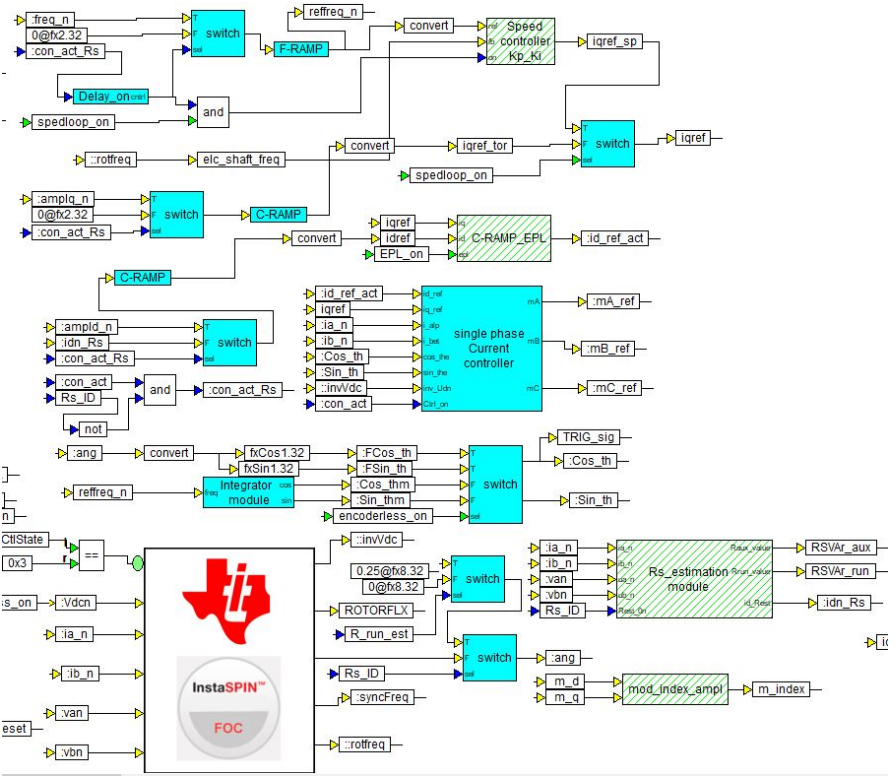


Fig. 7.30 Content 'Sph Drive Controller' module

voltage `Vdcn`, as used by the current controller.
Note that setting logic variable `encoderless_on` to zero, disables FOC secorless operation and this implies open loop current&frequency control will be active. However, the FAST®module will remain active under these conditions, hence an effective mode to test operation of this module.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational.

Operation of the drive (for the result shown in figure 7.31) is initially set with a speed reference of 2710 rpm (which is the rated speed) and the load controls the quadrature current amplitude (once steady state operation is achieved). A quadratic load speed curve is deployed, which is set to provide a torque of 0.634 Nm at 2710 rpm (which are the rated values of the machine). FAST®operation starts at 0.32 s and the torque is initially limited by the 1.3 A quadrature current setting of the speed controller. At time mark 1.5 s

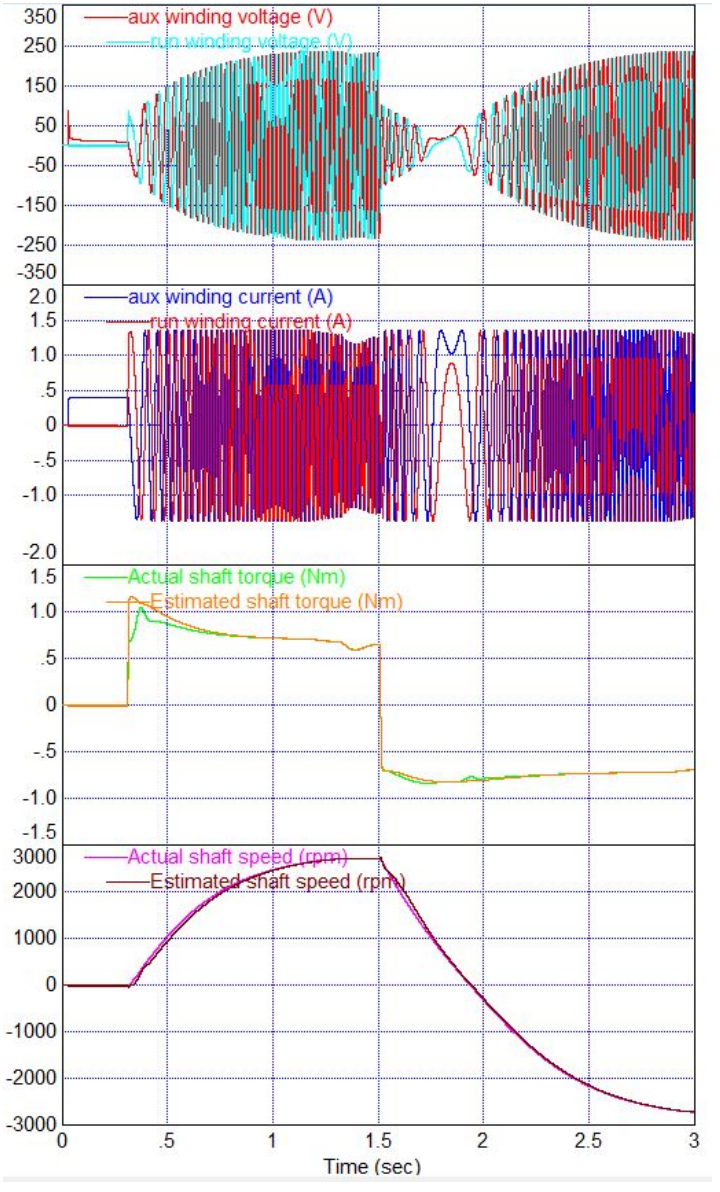


Fig. 7.31 PIL Result: FOC sensorless operation single phase IM drive, using FAST® with speed control

a shaft speed reference reversal from 2710 rpm to -2710 rpm is applied, as is apparent from figure 7.31).
The vector plot, which corresponds to time mark 3.0 s, is interesting, as it shows the scaled (by a factor 2) rotor flux vector ('green'), current vector ('red') and scaled (by a factor 1/50) terminal voltage vector ('blue').

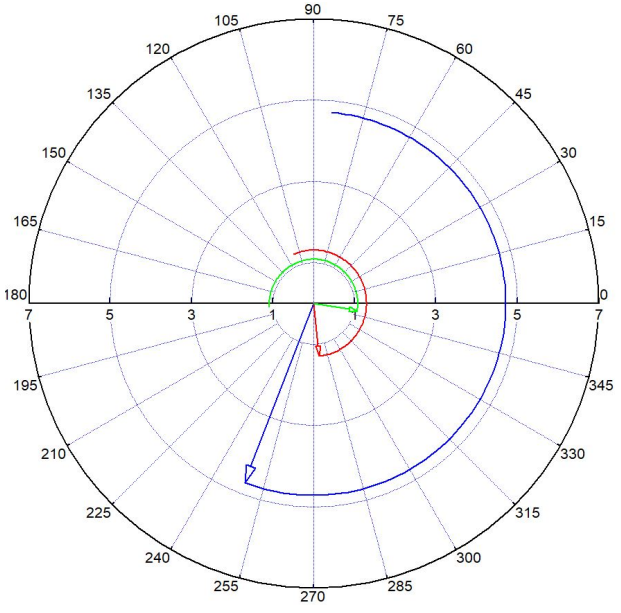


Fig. 7.32 PIL result, single phase IM drive operation, sensorless FOC-speed control active: vector plot

A key observation from this vector plot is the orientation of the current flux vector relative to the rotor flux vector. In this example the direct axis current setting is 0.4 A and the quadrature current is defined by the load, as speed control is active. Note that rotation is counter-clockwise as a negative reference speed of -2710 rpm is active.

Post analysis: Comparison between FOC sensorless and mains connected operation of a single phase IM

Reference file:

- `P-IM_sph_main.vsm`

The impact of using a single phase induction machine operating with a three phase converter and field-oriented sensorless control can be effectively illustrated by comparing operation of said machine connected to a unit[240]V/50 Hz supply. For this purpose the example IM [7] is considered and main connected operation, using the dual capacitor and centrifugal switch arrangement, as discussed earlier (see section 7.1) is given in figure 7.33. In this case an Embed model is shown with the same single phase motor IM model and mechanical load as used in the PIL model above (see figure 7.27).

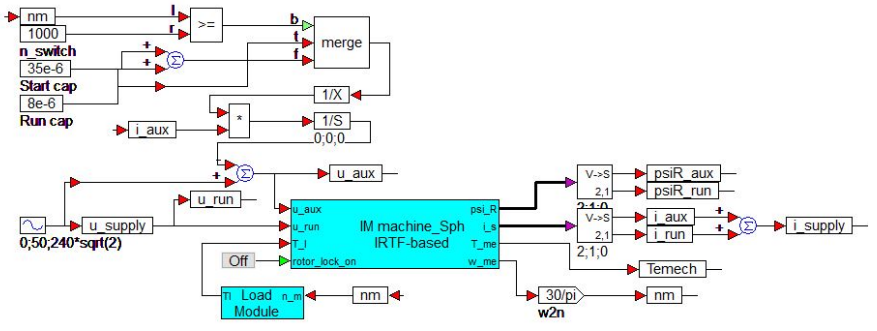
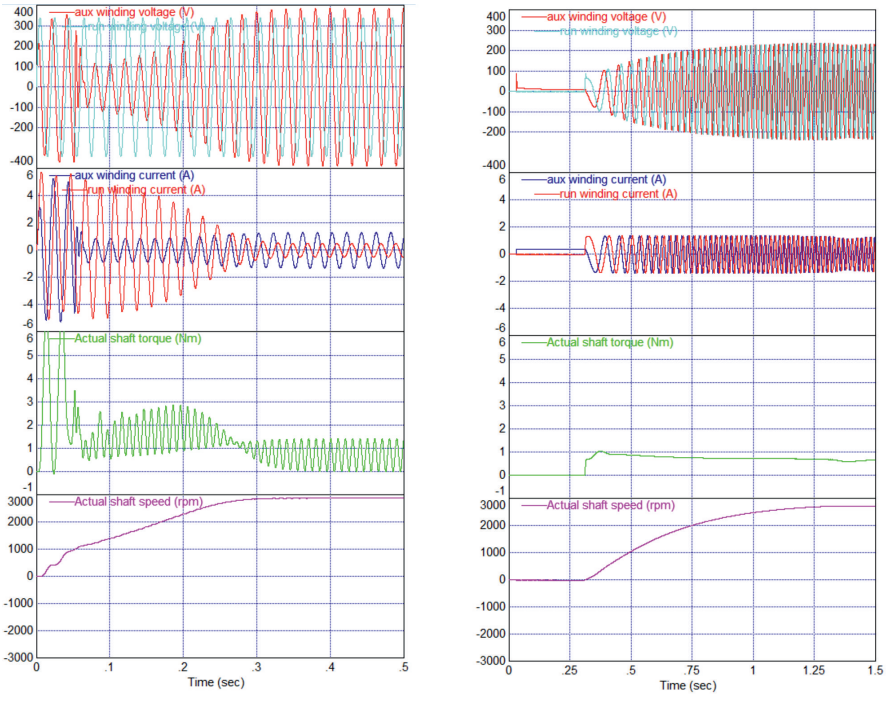


Fig. 7.33 Mains connected single phase example induction machine [7]

For the FOC drive, speed control is assumed with a speed reference setting of 2710 rpm (the rated speed of the example machine). The direct axis current reference value of 0.4 A has been judicially chosen to maximize the rotor flux, to a point where the converter modulation index approaches its limit value (as shown in figure 7.27). The mains connected IM makes use of a 35 uF 'start' and 8 uF 'run' capacitor. The 'start capacitor' is connected in parallel with the 'run' capacitor by the centrifugal switch unit until a shaft speed of 1000 rpm is reached, to improve the start-up torque of the machine.
The results given in figure 7.34 show the aux/run winding currents/voltages, shaft torque and shaft speed for the mains connected machine (figure 7.34(a)) and converter connected machine (figure 7.34(b)).
The vertical scales of both plots is the same, but the time scale is different as the main connected machine reaches the steady state shaft speed (which is the rated speed of 2710 rpm) faster then the converter connected machine. The reason for this is the shaft acceleration difference, that stems from the



(a) Mains connected start-up

(b) FOC sensorless start-up

Fig. 7.34 Comparison start-up performance single phase IM: Mains connected and FOC sensorless operation, using FAST®with speed control

fact that start-up torque levels are a factor x6 higher when compared with the converter connected machine (where the quadrature current is held to 1.3 A to stay within rated current conditions). Furthermore, observation of figure 7.34 also shows the presence of a large torque ripple component in the mains connected machine during steady-state operation. Overall current levels of the FOC converter controlled machine are significantly lower and excellent torque control is achievable. Furthermore, speed reversal is also possible, as was shown earlier (see figure 7.31) with the same machine and drive settings.

7.3.1.2 IM Sph FAST FOC-F069-HVkit: COSIM

Reference files:

- `IM_Sph-FAST-069_HVkit-COSIM.vsm`
- `Sph-IM_FAST-PIL_mod.vsm`
- `HVkit_INVv1.psimsch`
- `my motor model_Sph.psimsch`

To evaluate operation of a controller with a switching converter, in this case the Texas Instruments `HV kit`, use can be made of a `PSIMCoupler` module, as indicated in figure 7.35. This approach implies the use of PSIM together with Embed, which is referred to as 'co-simulation'. Furthermore, a 'processor in the loop' (PIL) approach is needed given the use of a classified sensorless control algorithm FAST®. The `IM-Sp-FAST-PIL` module is shown hatched in this

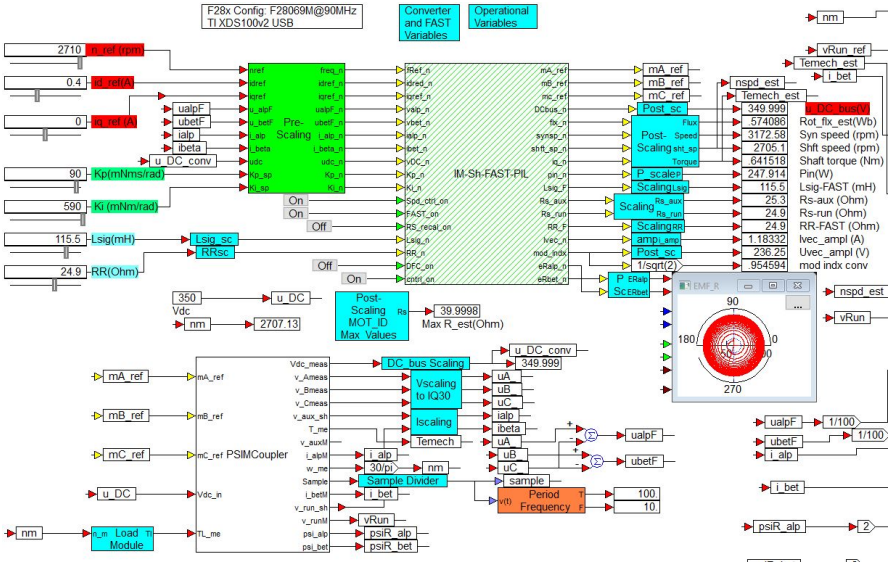


Fig. 7.35 COSIM based single-phase IM drive, Field-oriented torque/speed control using FAST®

case, as its sampling frequency is locally set to 10 kHz, whereas the 'system' frequency (set in the 'system properties' dialogue box) is set to 20 MHz. The latter is much higher, given the need to accurately represent 10 kHz voltage switching waveforms generated by the PWM unit of the converter. The content of the `IM-SP-FAST-PIL` module was introduced earlier, see figure 7.29. However, the current and voltage measurements now make use of the signal processing circuitry connected to the half-bridges.

In this case the auxiliary (aux) and run winding currents are measured using shunts, as was discussed earlier (see figure 7.15). After sampling and signal-processing in the converter, these variables appear as: `ialp`, `ibeta` in figure 7.35. The low pass filtered voltages (which removes the PWM component) of the three half-bridges is also present in figure 7.35 and are also scaled and sampled by the module `Vscaling to IQ30`. In this case the per unit variables are rescaled to actual values: `u_A`, `u_B` and `u_C` respectively. For single phase machines use is made of `SUM` modules to find the LPF voltages

across the auxiliary (aux) and run windings, defined as `ualpF` and `ubetF` respectively.
The 'Converter and FAST Variables' dialogue box (see figure 7.35), defines the scaling used by the Analogue to Digital converter, which has a maximum input value of 3.3 V and is directly linked to the signal processing undertaken for voltages and currents as discussed above. For example, the 'ADC-fullscale-voltage' value is the theoretical peak-to peak input voltage which must be applied to the low-pass filter network (see figure 7.15) to ensure that the ADC input level varies between $0 \rightarrow 3.3$V. The current shunt voltage(s) `v_aux_sh`, `v_run_sh` are sampled by the `Iscaling` module shown in figure 7.15. When dealing with the actual MCU, the ADC converter also takes over this sampling process. Apart from the measured voltage/currents generated by the converter, the PSIM coupler unit also provides the actual motor model aux/run current vector: `i_alp`, `i_bet` (in stationary coordinates), 'run' winding voltage and shaft speed, which are presented by the variables `vRun`, `i_alp`, `i_bet` and `n_m` (after conversion to rpm) respectively, see figure 7.35.

The single phase induction machine model `IM` used in the PSIM coupler module (see figure 7.15), is based on the IRTF model discussed earlier (see figure 7.16).
Results achieved with the COSIM approach are shown in figure 7.36, using a quadratic load-speed characteristic, which generates 0.634 Nm at 2710 rpm (these are the rated values for the example single phase machine [7] used).
Indicated in said figure is the 'run'winding voltage waveform together with the reference voltage, 'run' winding current, shaft torque (actual and estimated) and shaft speed (actual and estimated). Note that FAST®is activated around time mark 0.03 s.
Note that drive operation at time mark 1.5 s is very close to the rated speed and the direct axis reference current was judicially chosen to achieve operation at the maximum controller modulation value of 1.35 . The corresponding modulation index for the three-phase converter is a factor $1/\sqrt{2}$ lower, as is also shown in figure 7.35.
For illustrative purposes a 'zoom in' of figure 7.36 around the 1.5 s time mark is shown in figure 7.37. These results clearly show the presence of the 10 kHz PWM voltage, which for single phase machines has an amplitude of ± 350 V (which is the DC supply voltage used). Furthermore, estimated and actual shaft speed/torque are in agreement (apart from the initial start-up 0.375 mark). Also observation of the actual shaft torque learns that there is no significant torque ripple when operating the single-phase machine with a three-phase converter. This is in stark contrast with mains connected operation of said motor, as was discussed earlier (see figure 7.34).
The vector plot (see figure 7.38), which corresponds to time mark 1.5 s, is interesting, as it shows the scaled (by a factor 2) rotor flux vector ('green'), current vector ('red') and scaled (by a factor 1/100) terminal voltage vector ('blue').

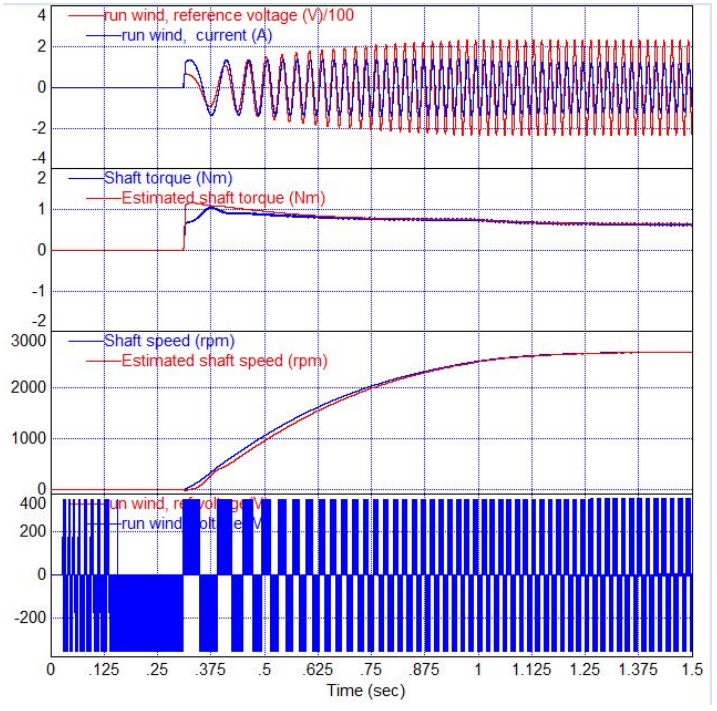


Fig. 7.36 Result COSIM, single-phase IM Field-oriented torque/speed control using FAST®

A key observation from this vector plot is the orientation of the current vector relative to the rotor flux vector. In this example the direct axis current setting is 0.4 A and the quadrature current is defined by the load, as speed control is active. Note that rotation is clockwise as a positive reference shaft speed of 2710 rpm is active. Furthermore, the current vector locus has a low PWM switching component [13], which explains the low torque ripple component of the drive.

Co-simulation, as discussed in this sub-section, is instructive for understanding the impact of using a switching converter on drive operation,e.g to asses the impact of shaft torque ripple due to PWM.

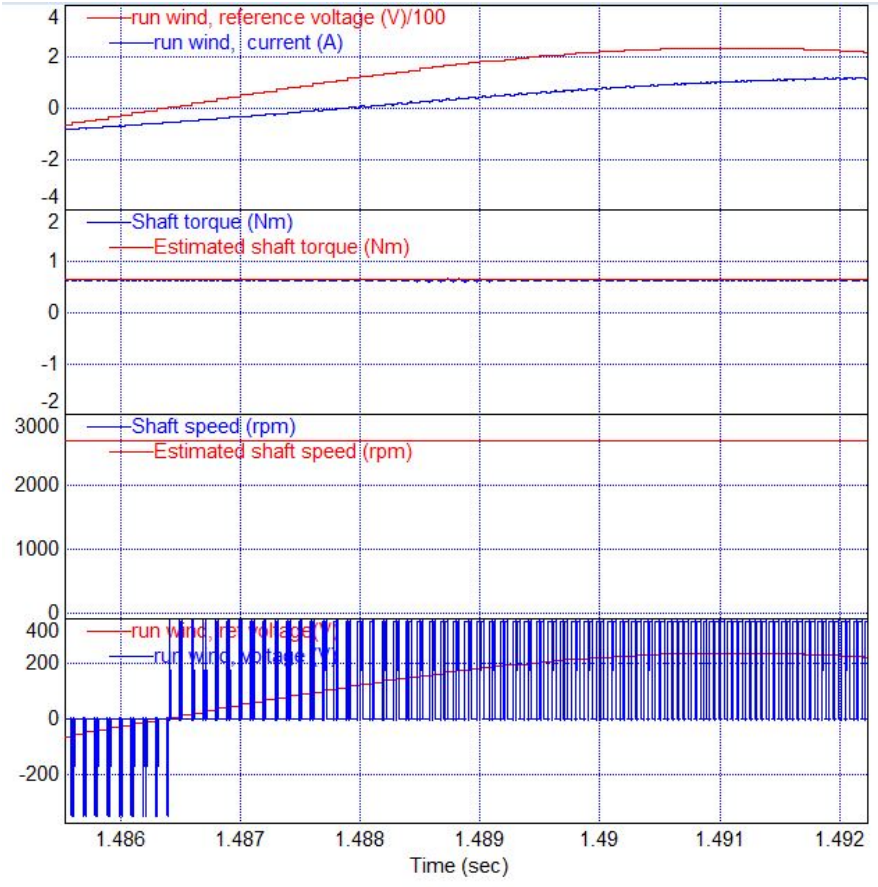


Fig. 7.37 Result COSIM:'zoom in', single-phase IM Field-oriented torque/speed control using FAST®

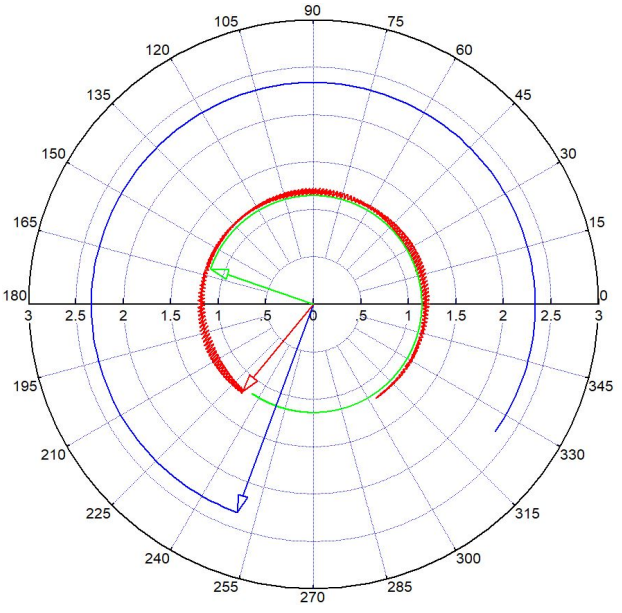


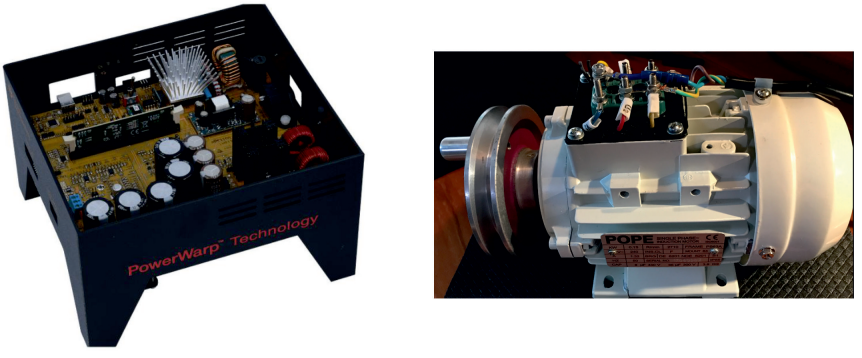
Fig. 7.38 PIL-COSIM result, single phase IM drive operation, sensorless FOC-speed control active: vector plot

7.3.1.3 IM Sph FAST FOC-F069-HVkit: HIL

Reference files:

- `IM_sp_FAST_F069_HV-compile.vsm`, for HIL-compile
- `IM_sp_FAST_F069_HV-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 7.39 represents the `HV kit` converter (with ISO `F28069` card), which is connected to a single phase induction machine[7]. Said machine (initially configured for single phase mains operation) was reconfigured (by removing the stationary part of the centrifugal switch and two capacitors) to operate with a three-phase converter, as shown in figure 7.39(a).



(a) Texas Instruments HV kit (b) POPE Single -phase 0.18 kW IM

Fig. 7.39 MCU-converter setup for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs, which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 7.40 shows the 'FOC sensorless control single phase IM.. ' module, which must be compiled to generate an out-file.
Inputs, shown in the stage as constants (for readability purposes) for this controller module are the user direct/quadrature axis reference currents `id`, `iq` and the reference shaft speed `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller. Furthermore, two inputs `Lsig_pu`, `RR_pu` are provided to allow the user to set the leakage

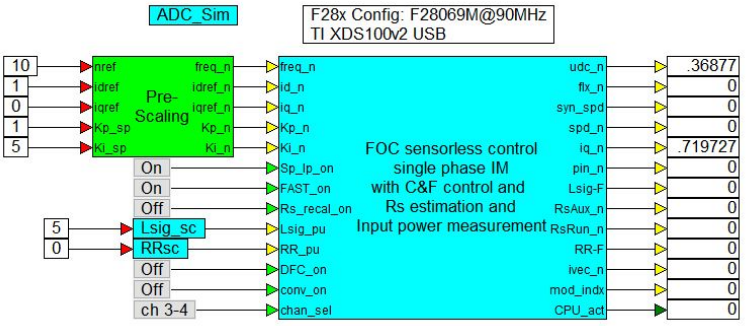


Fig. 7.40 HIL-Compile model of a single phase Induction machine drive with field-oriented control and FAST®

inductance L_σ and rotor resistance R_R (via sliders) respectively. Six logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. Three logic inputs are used to select field-oriented control (FOC) operation with `FAST_on` (algorithm FAST®active), speed control operation with FOC `Sp_lp_on`, and `DFC_on`, as discussed earlier for three-phase machines (see section 6.3). The logic input: `RS_recal_on` estimates the 'auxiliary' (aux) and 'run' winding resistances at start-up (when the button `conv_on`) is activated. Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n`, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- The per unit rotor flux estimate `flx_n`.
- A per unit synchronous speed estimate `syn_spd`.
- A per unit shaft speed estimate `sp_n` .
- The measured per unit quadrature current `iq_n`, used to estimate the shaft torque.
- A per unit estimate of the motor input power `pin_n`. This is also useful as an estimate for the drive input power (but this ignored the converter losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx {}^{p_{\mathrm{in}}}/_{u_{\mathrm{DC}}}$.
- The leakage inductance used by FAST®`LsigF`.
- The measured 'aux'winding resistance (if activated at start-up) `RsAux_n` (duration set in the `100 Hz` module with unit `Timer-Rs`).
- The measured 'run' winding resistance (if activated at start-up) `RsRun_n` (duration set in the `100 Hz` module with unit `Timer-Rs`).
- The rotor resistance value used by FAST®`RR-F`.
- The measured per unit current vector amplitude `ivec_n`.
- The controller modulation index `mod_indx`.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 7.40 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 7.41.

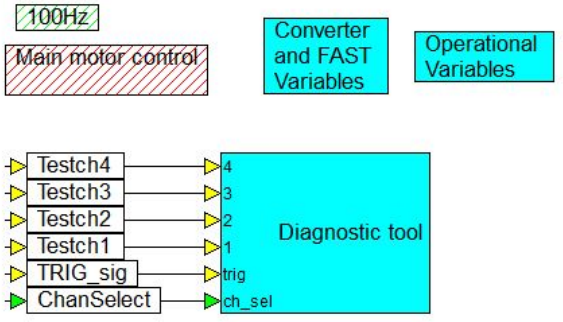


Fig. 7.41 One level down into the 'FOC sensorless control ...' module

Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'operational Variables' dialogue box (see figure 7.42), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference currents/frequency. In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the gain/bandwidth setting are introduced together with the variable `Res_est_current` which defines the DC injection current amplitude used for resistance estimation at start-up (if selected).

The 'Converter and FAST Variables' dialogue box (see figure 7.43), defines the scaling used by the Analogue to Digital converter. This module also shows the low-pass filter corner frequency `Voltage_filter (Hz)` used by the converter for measurement of the half-bridge voltages (see figure 7.15).
Furthermore, the speed controller pole `Speed filter pole (Hz)` is introduced, which is a LPF of the estimated shaft speed, set here to 50 Hz. Also added to this dialogue module are motor parameters: motor pole pair number p, rotor flux `psiR (Wb)`, aux/run winding ratio, aux/run winding resistance estimates and rotor resistance R_R of a six parameter single phase IRTF induction machine model. Note that the rotor flux estimate used by FAST®is tied to the reference direct axis current setting.

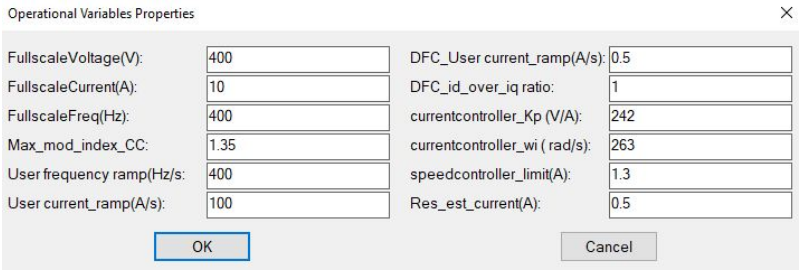


Fig. 7.42 HIL: Dialogue box 'Operational Variables'

In the sequel of this HIL a simple approach to determining the rotor resistance R_R will be discussed. This dialog box also requires a `R_shift` value, which must be set to ensure that the module `POST SCALING Motor ID` (see figure 7.45) shows a resistance value that is larger then the measured aux/run winding resistances.
This dialog box also sets the variable `F_est/F_sampling`, which in effect sets the sampling frequency used by FAST®, relative to the controller sampling frequency. Typically these two are selected to be equal, i.e in this case 10 kHz. The sampling frequency for the speed controller is set by the variable `f_speed/F_sampling` and in this case is 1 kHz. The variables shown in this dialogue box are the 'primary variables' required for FAST®. However, in the `100 Hz` module a set of 'secondary variables' are defined, which do not (typically) need to be changed.

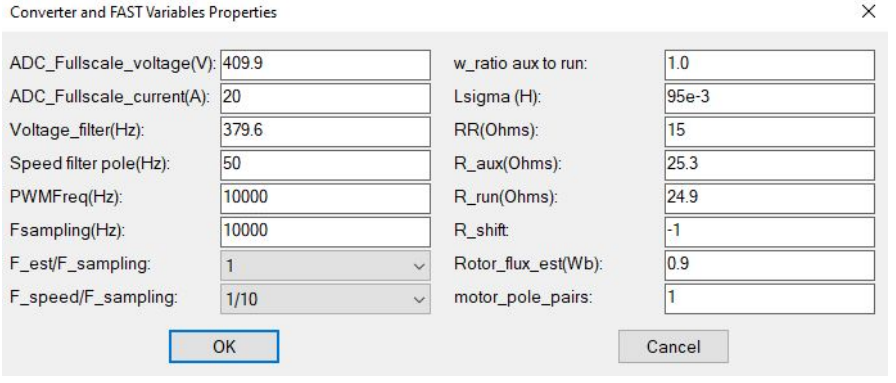


Fig. 7.43 HIL: Dialogue box 'Converter and FAST Variables'

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1`, `Test2` or `Test3`, `Test4` by using a 'channel select' button discussed above.

This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 7.41 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 7.44. Of these, the 'Sph Drive Controller' has been discussed earlier, see figure 7.30 where it was used in phases : PIL and COSIM of the development tool-chain.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 10 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 2 μs
- Board in use: `F28069 ISO` control card with `TI HV kit` based converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.
The `ADC-PWM-TI_HV` unit is used generate the PWM signals for the converter. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to aux/run voltages: `u_alp`, `u_bet`. The two measured (per unit) aux/run currents are represented by outputs `i_alp`, `i_bet` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the `100 Hz` module). Estimation of the ADC offset's is also controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 7.45 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. Three sliders are used to set the direct/quadrature axis current reference values and the shaft speed. Also shown are two sliders that set the proportional and integral gain of the speed controller. Furthermore, two

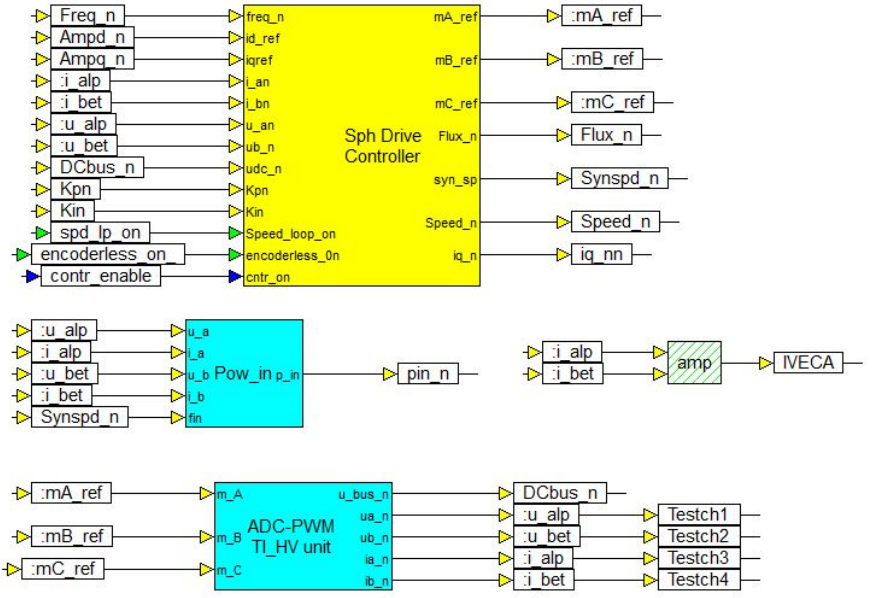


Fig. 7.44 One level down into the 'Main motor controller' module

sliders are added, which set the rotor resistance and leakage inductance used by FAST®(these overide the dialog values R_R, L_σ shown in figure 7.43). Post-scaling modules are used to convert the output variables shown in figure 7.40 to actual values, as shown with numeric displays. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.
The plot given in figure 7.45 displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit EMF $\vec{e}_R$ (see equation 7.3b) vector locus. The shaft speed reference in figure 7.45 is set to 2710 rpm. These results shows operation under FOC-speed control with the machine operating under no-load conditions. An instructive exercise for the user is to switch operation to FOC 'torque control' (speed loop OFF) with the direct axis current set to 0.4 A and quadrature current reference initially set to zero. Then hold the rotor by 'hand' and feel the effect of torque application when moving the quadrature reference slider.

In addition, a DC current probe with 'PicoScope' [20] was used to record the 'aux' current when using the speed reference reversal option (see figure 7.45), which causes a reference speed change from (rated speed) $2710 \rightarrow -2710$ rpm. Such a reference speed step change will cause a current change similar to the PIL result (see figure 6.62) shown earlier, as may be observed from figure 7.47.

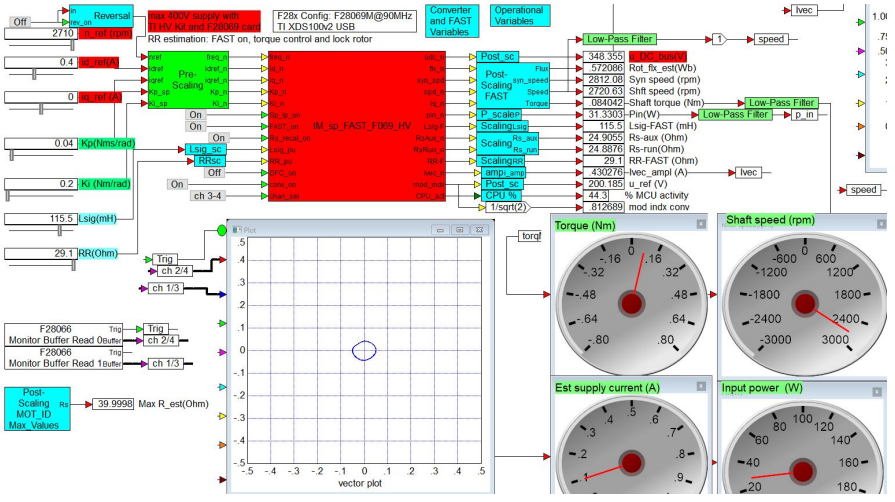


Fig. 7.45 HIL-Debug model, F069-HV KIT of a single phase induction machine drive with field-oriented control using FAST®

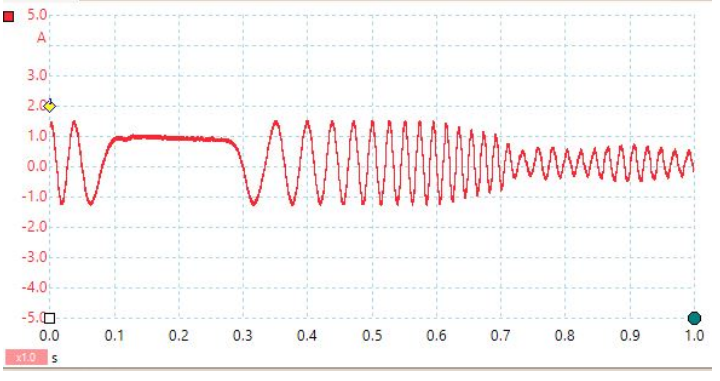


Fig. 7.46 Measured auxilliary winding current, F069-HV KIT of a single phase induction machine drive with field-oriented control using FAST®: rated speed 2710 → −2710 rpm reversal

This result, again confirms the versatility of sensorless FOC control that can be achieved with a single phase machine when connected to a three-phase converter.

Post analysis using FAST®for single phase IM

The application of field-oriented control for single phase induction machine drives requires knowledge of several critical parameters of the machine and

this issue is raised in the 'Post-analysis' section here.
An important parameter to consider is the leakage inductance given the fact that the voltage across said inductance is substantial, given the ratio of controller current amplitude to equivalent short-circuit current for this machine. This ratio is equal to ≈0.3 for this machine, hence the voltage across the inductance is approximately said value times the EMF. Measurement of the leakage inductance using an L-C-R meter provides a ball park figure. Further fine tuning can be done by running the drive sensorless and under speed control, no-load and then adding a (in this case) 'hand induced mechanical load', which basically adds a quadrature current component to the current. This effect is shown in figure 7.47 which is a plot also provided in the HIL-debug/run file (see figure 7.45) and gives the vector current amplitude and the rotor flux normalised to the rotor flux value set in the dialogue box (see figure 7.43). Note that this variable will change if the direct axis current reference is varied.

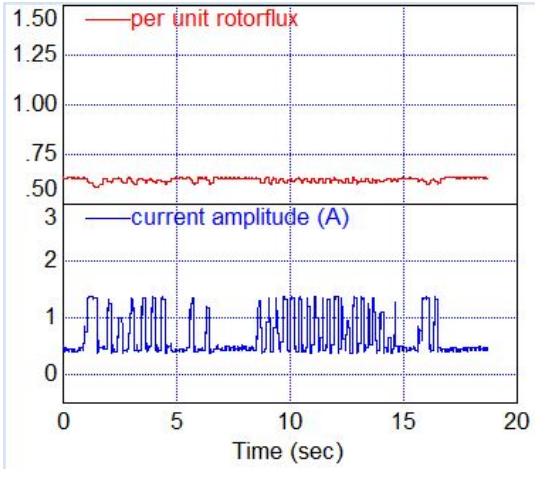


Fig. 7.47 HIL-Debug model, Tuning the leakage inductance used by FAST®

Varying the vector current amplitude as shown will affect the EMF (and therefore the rotor flux amplitude) used by FAST®if the leakage inductance does not match the value experienced by the machine. Consequently, the user should incrementally change the leakage inductance slider to minimize variations of the scaled flux when machine load is applied.

The second issue to consider is the rotor resistance. A simple approach to determining this value is to run the machine under sensorless torque control (speed-loop OFF) AND lock the rotor shaft. The latter implies that the estimated shaft speed MUST be zero as can, for example, be observed from the shaft speed gauge. The relationship between synchronous speed and

shaft speed (see equation (7.4)) shows the importance of the rotor resistance. Hence with the rotor physically locked the rotor resistance slider should be incrementally changed until the shaft speed gauge reads zero. The synchronous speed and quadrature current required for said expression are provided by the controller.

Another important issue to consider is the magnetization characteristic of the machine as introduced earlier (see figure 7.26). This characteristic can be found by run operating the machine at, for example, 50% rated speed, under sensorless speed control and incrementally varying the direct axis reference current from typically, 50 % rated to 150 % and recording the corresponding rotor flux values, as shown in figure 7.26 for the machine used in this example. The magnetizing characteristic is important as it provides the user with indication of saturation levels in the machine, relative to the choice of 'rated' direct axis current used. Furthermore, the linear approximation of the magnetizing characteristic is used in the single phase IRTF based IM models, as discussed earlier in the SIM and PIL examples.

Finally, for single phase machine the parameter k is required, which is the ratio between auxiliary and run windings, in terms of the number of turns. For this machine, the ratio is set to ONE and a good tuning tool is observation of the locus tied to the EMF vector $\vec{e}_R$ (see equation (7.3b)), which is a function of said parameter.

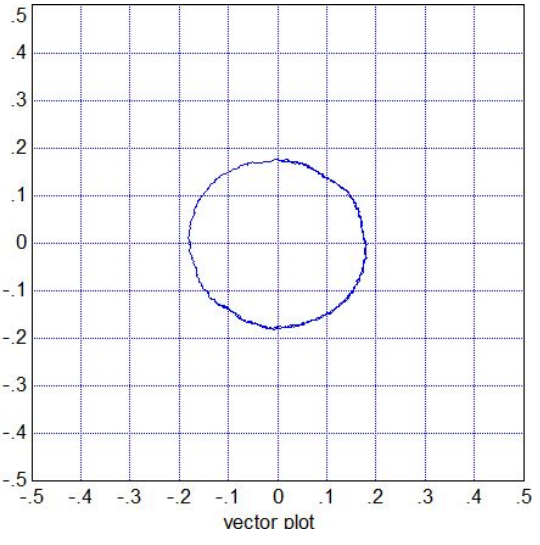


Fig. 7.48 HIL-Debug model, Checking the aux/run winding ratio

The tuning process is a matter of achieving a concentric EMF locus, as shown

in figure 7.48. This process can also be considered, by for example using the PIL model (see figure 7.27) and changing the aux/run winding ratio in the IRTF model first and then observing the EMF locus. Then secondly, tuning can be implemented by changing the value of k in the controller dialog box and observing the EMF locus, i.e to achieve a concentric circle.

Chapter 8
Dual drive applications sensored

A dual electrical drive, consist of two electrical machines of which the respective rotors are coupled together, as shown in figure 8.1. The (assumed) primary machine with inertia J_1 defines the shaft speed $\omega_{\text{me}} = \omega_{\text{me1}} = \omega_{\text{me2}}$ as the rotors are rigidly coupled together. The rotor of the secondary machine with

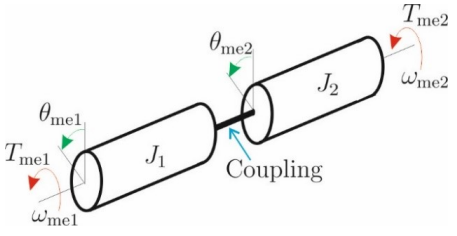


Fig. 8.1 Dual drive torque/speed variables convention

inertia J_2 can be the same type of machine as the primary, or different and either three or single phase , i.e brushed DC. However, the secondary machine can be simply a mechanical load with shaft torque that usually opposes the shaft torque generated by the primary machine. If the primary machine is configured as a motor the product $T_{\text{me1}}\,\omega_{\text{me1}}$ is POSITIVE. If the secondary machine is set to operate as a generator (or more commonly referred to as a "dynamometer" the product $T_{\text{me2}}\,\omega_{\text{me2}}$ is NEGATIVE. The directional speed is the same, hence motor/dyno operation is dictated by the polarity of shaft torque as set by both controllers of a dual drive.
The dual electrical drive shown in figure 8.2, consists of the processor (MCU) that houses the control algorithms and all the functionality needed to use the measured current/voltages and generate the PWM signals for both converter modules. Phase currents are measured using in-line sensors and low-pass filters (LPF) are used to measure the converter output voltages and eliminate the PWM component. The 12 bit ADC unit digitizes the incoming voltages/currents of both converters. The term 'sensored' dual drive, underlines

the fact that at least one of the two machines deployed has an encoder that can be used as required. In this case it is the PM machine[2], that has an encoder. Two controllers are used and the algorithm deployed in each, depends

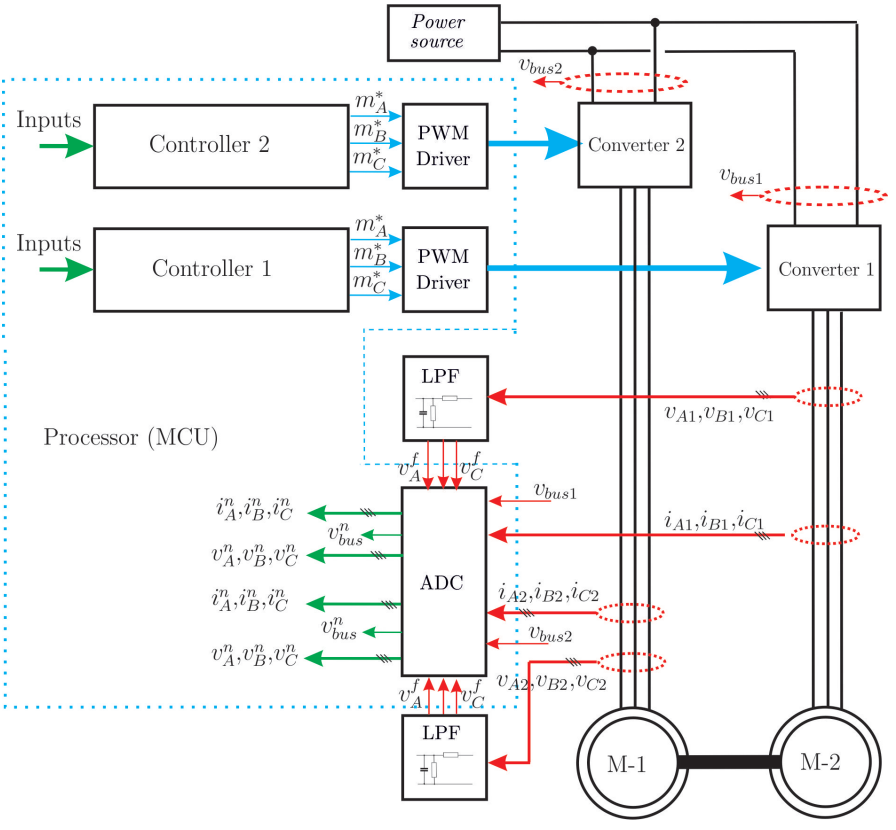


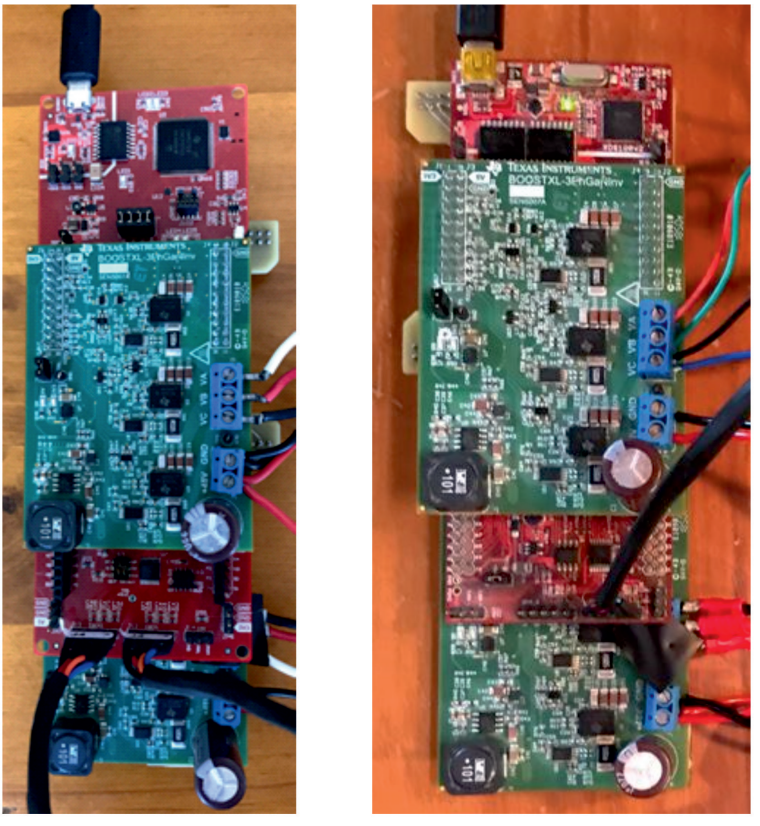
Fig. 8.2 General dual drive concept

on the machines used and in this chapter the following configurations are considered:

- DC-PM: Brushed DC machine [25] connected to a sensored PMsyn machine [2].
- PM-PM: Sensored PMsyn machine [2] connected to a sensored PMsyn machine [2].
- IM-PM: Induction machine [25] connected to a sensored PMsyn machine [2].
- HbSt-PM: Sensorless Hybrid Stepper machine [9] connected to a sensored PMsyn machine [2].

For the first three cases the development tool-chain components: SIM and HIL will be discussed, with a hardware set-up that consists of a Texas

Instruments LAUNCHXL-F280049C MCU with two BOOSTXL-3PHGANINV 12-60 V/10 A converters (see figure 8.3(a)). The fourth case 'HbSt-PM', is an advanced dual drive example, which makes use of the development toolchain components: PIL and HIL, with a hardware set-up that consists of a Texas Instruments LAUNCHXL-F28069 MCU with two BOOSTXL-3PHGANINV 12-60 V/10 A (see figure 8.3(b)).



(a) F280049 with two GANINV modules (b) F28069 with two GANINV modules

Fig. 8.3 Dual drive hardware for four cases considered in this chapter

When considering dual drive operation the user needs to be aware that the concept of a machine with static load is now replaced by a dynamic 'dual machine' topology, where one machine acts as the 'primary' (configured for motoring) and the other as 'secondary' (configured for generating). Correspondingly, the energy flow is dynamic, pending the mode of operation

deployed.
A salient advantage of 'dual control' is the ability to deploy 'remote learning' as shown in figure 8.4 where a student's laptop linked via internet to the instructor's computer can control a dual drive. This approach is advantageous

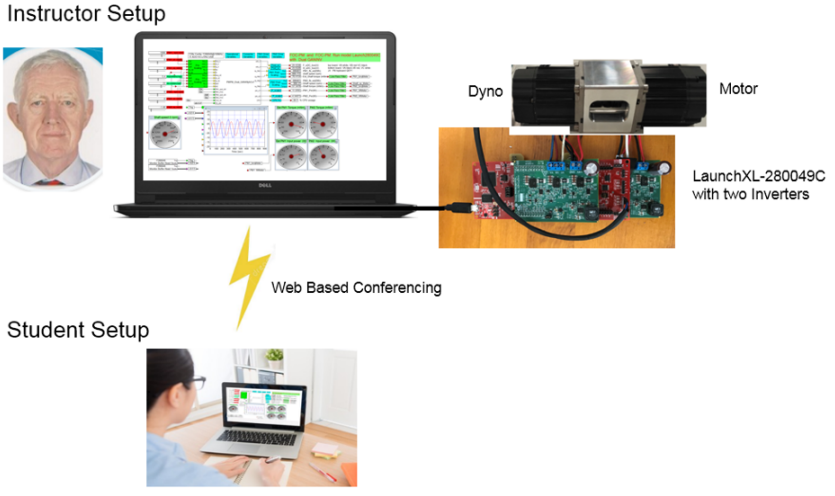


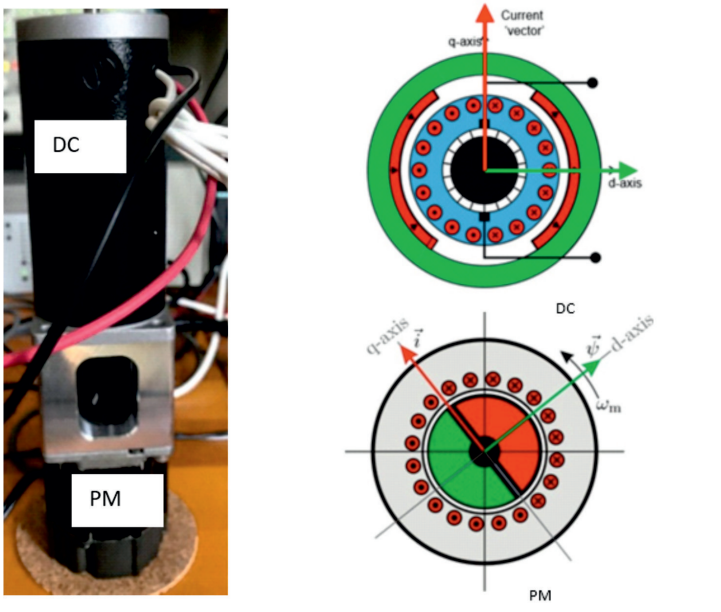
Fig. 8.4 Use of dual drives for remote learning

as it advocates the remote learning process in the field of electrical drives. Hence, this implies that a student can, for example, set up drive control for the primary machine (acting as a motor) and also manipulate the load torque using the secondary machine (acting as a 'dyno'), without the need to be physically present near the hardware set-up.

8.1 Dual drive sensored DC-PM

The dual drive considered in this section consists of a brushed DC machine [25] mechanically connected to a PM synchronous machine [2] as shown in figure 8.5. Included in this figure are two models [13] of said machines, which shows the importance of controlling the current vector relative to the 'dq' coordinate system(s) in order to control the shaft torque. For the brushed DC machine this coordinate system is tied to the stator and the brush/commutator is used to produce a stationary current vector that interacts with the PM flux vector to produce torque. This situation is reversed in the PM synchronous machine, which has a rotating flux vector, hence a rotating orthogonal (to the flux vector) current vector is required that is generated in the three-phase distributed winding [28]. The PM machine is provided with an incremental

encoder, which enables sensored field-oriented control for the PM machine and speed control of the DC machine. The overall drive concept of the DC-



(a) DC- PM Drive (b) Brushed DC and PMsyn machines

Fig. 8.5 DC-PM dual drive setup

PM drive, as given in figure 8.6 shows the two controllers that generate the modulation indices for the respective converters that are connected to the machines in question. An ADC unit is used to provide the currents, LPF converter voltages and encoder signals required for the two controllers. For the DC machine: voltage, current and speed control as discussed earlier (see sections 3.1.1, 3.2.1 and 3.3.1) is implemented in 'controller 1'. Control for the PM synchronous machine is implemented in 'controller 2' and consists of: sensored field-oriented current and speed control (see section 4.3.1). In addition, open loop current & frequency control (see section 4.2.1) , can be enabled if required, e.g to check alignment of the encoder angle with the flux vector.

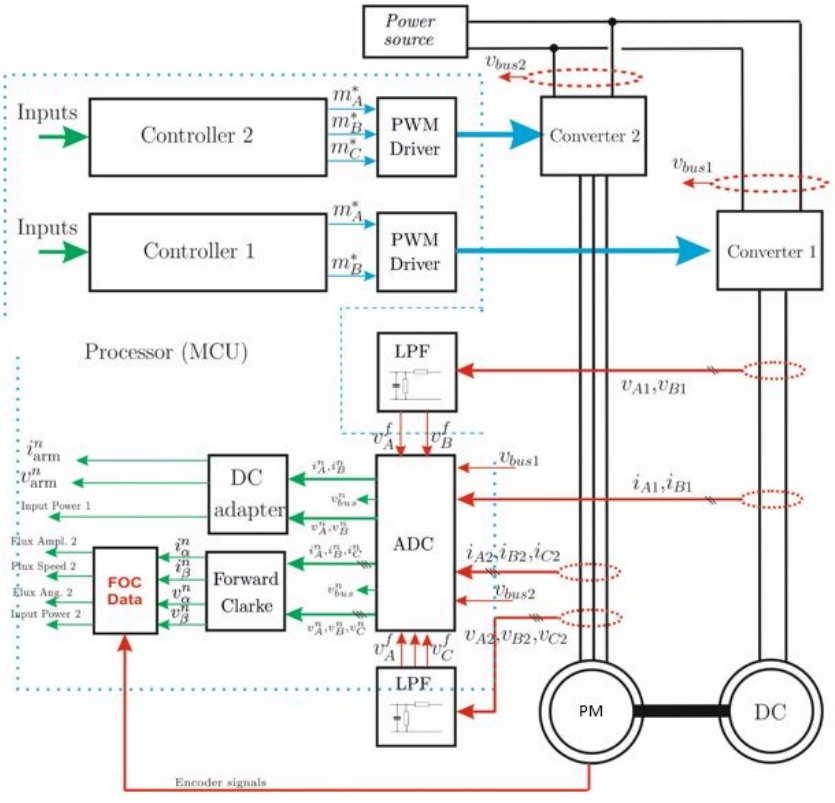


Fig. 8.6 DC-PM dual drive concept

8.1.1 Dual drive sensored DC-PM:SIM

Reference file: `DC-PM_0049_GAN-SIM.vsm`.

This example considers the simulation of the dual PMsyn Teknic machine [2]-DC machine [25] connected to a Texas Instruments F0049 twin GANINV converter setup, as represented in figure 8.7. The aim is to examine operation (for different operating modes) of the drive using two fixed point controllers, that will subsequently be used in the HIL implementation.

Drilling into the `DC_current/armV controller & spd controller` reveals

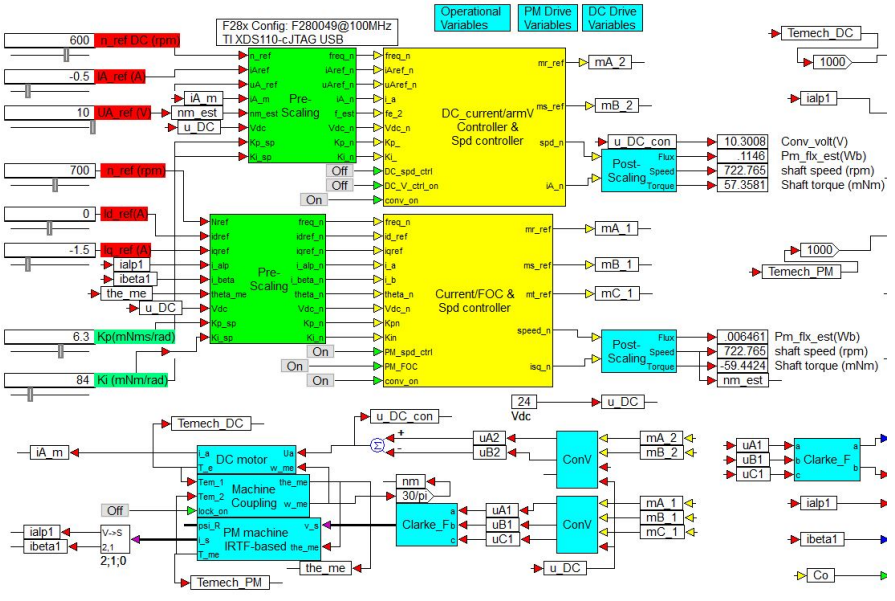


Fig. 8.7 DC-PM dual drive concept: simulation model

the controller structure shown in figure 8.8. Outputs of the controller are two modulation indices which control half-bridges 'A' and 'B' ('C' is not used for the DC machine). The voltages supplied to the motor `u_A2` and `u_B2` are found by multiplying the modulation indices `mA_2`, `mB_` by half the DC bus voltage as executed in module `Conv`, connected to the `DC motor` module, see figure 8.8.

The IRTF based PM model [13] `PM machine IRTF based` as introduced earlier (see figure 2.12) is connected to an 'average voltage per sample' converter `Conv` that, in turn is linked to the module `current/FOC & Spd controller` which is similar to one introduced earlier (see figure 4.177).

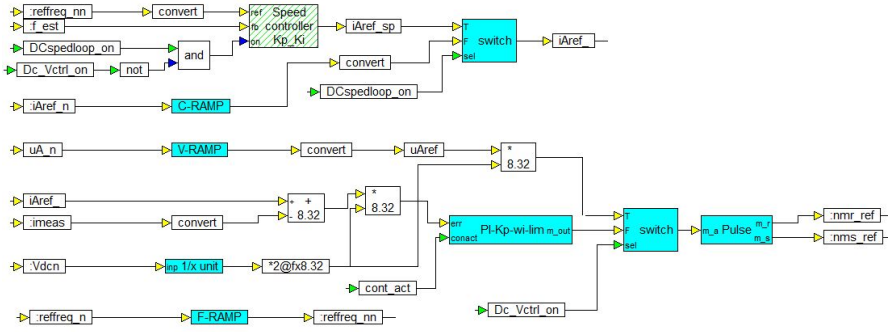


Fig. 8.8 DC controller structure

Both machine models are connected to a module `Machine Coupling` which represent a non-flexible coupling that is represented by expression (8.1).

$$T_{\text{me}}^{\text{DC}} + T_{\text{me}}^{\text{PM}} = J_c \frac{d\omega_{\text{me}}}{dt} \tag{8.1a}$$

$$\omega_{\text{me}} = \frac{d\theta_{\text{me}}}{dt} \tag{8.1b}$$

where J_c represents the combined inertia of both machines. The variables ω_{me}, θ_{me} represent the rotational shaft speed (rad/s) and shaft angle (rad) respectively.

For DC machine control three sliders: `n_refDC`(DC speed control), `IA_ref` (current/torque control), `UA_ref`(armature voltage control) are available, pending the operation mode selected, as will be discussed shortly. The PM machine controller also makes use of three sliders: `n_ref`(PM speed control), `id_ref`(direct axis current control), `iq_ref`(quadrature axis current control), as required for FOC-speed/torque control.
In addition, two sliders: `K_p`, `K_i` are present, which set the proportional and integral gain values for both DC/PM speed controllers (of which only one is used at any time).

A total of six buttons are connected to these controllers, which define the operation modes of the dual drive. In the ensuing subsections the relevant operation modes are discussed together with the results obtained.

8.1.1.1 DC-Voltage control, PM-Torque control: SIM

The operational model considered here requires the following controller button settings:

- DC: `DC_spd_ctrl`: OFF
- DC: `DC_V_ctrl_on`: ON
- DC: `conv_on`: ON
- PM: `PM_spd_ctrl`: OFF
- PM: `PM_FOC`: ON
- PM: `conv_on`: ON

With these button settings, armature voltage control for the brushed DC machine is enabled together with sensored field-oriented torque control of the PMsyn machine. This implies that the following set of users sliders are active:

- `UA_ref(V)`: Control of the DC brushed motor armature voltage.
- `Id_ref(A)`: Control of the PM direct axis current. Typically, this setting should be set to zero, as torque is controlled by the quadrature component.
- `Iq_ref(A)`: Control of the PM quadrature axis current, which in turn controls the shaft torque.

An example of simulated drive operation, given in figure 8.9 is based on a armature voltage slider setting of 10 V and quadrature slider reference of -1.5 A (the direct axis reference current is held at 0 A). The armature voltage ramp setting is equal to 100 V/s, hence at start up the armature voltage increases until its steady-state value is reached at time mark 0.1 s. The current ramp setting is 50 A/s and this causes a PM machine torque build-up to its steady state value at time mark 0.03 s as may be observed from figure 8.9, At start-up the drive accelerates because the DC brushed machine torque is greater then the PM load torque. The steady-state shaft speed is reached when the DC torque matches the PM machine torque in amplitude but opposite in polarity. For the conditions shown here, energy flow is from the DC machine to the PM machine, i.e this unit functions as a generator. If for example, the quadrature slider reference is reversed (and maintaining the same armature voltage) to 1.5 A, the energy flow will be reversed, i.e the DC machine will then act as a generator and the PM machine as a motor. The shaft speed will also increase, but will remain positive.
The vector plot given in figure 8.10 corresponds to time mark 0.2 s for the simulation shown in figure 8.9 and confirms PM operation in generating mode (quadrature current component lagging the d-axis which is tied to the PM flux). Shown in figure 8.10 are the vectors that represent the direct axis ('green'), current vector ('red') and terminal voltage vector ('blue').

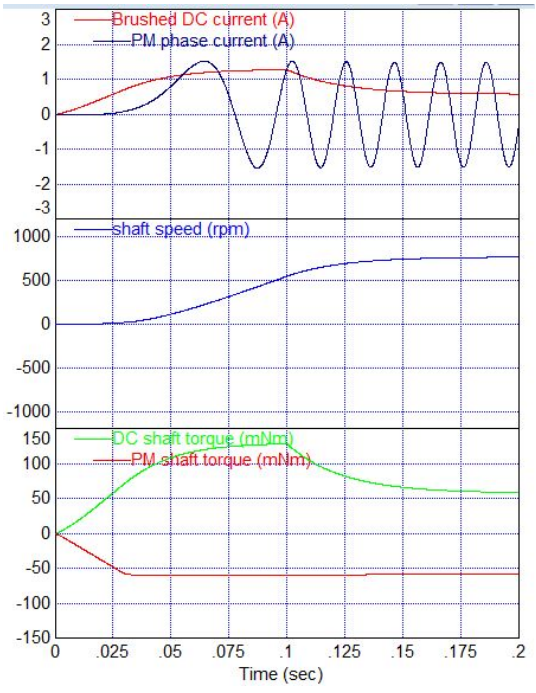


Fig. 8.9 Results: DC-voltage/PM torque control: SIM

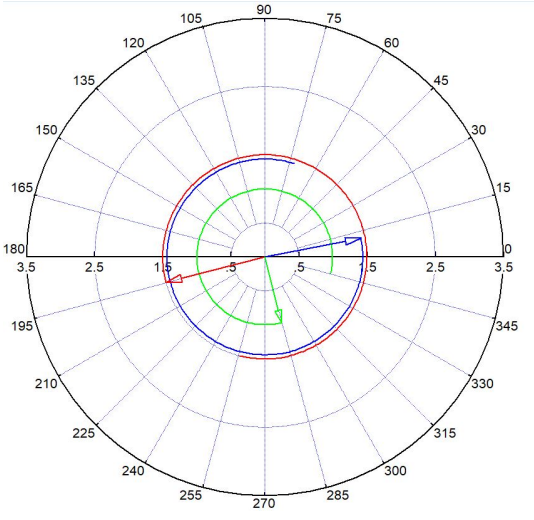


Fig. 8.10 Results: DC-voltage/PM torque control, vector plot:SIM

8.1.1.2 DC-Speed control, PM-Torque control: SIM

The operational model considered here requires the following controller button settings:

- DC: `DC_spd_ctrl`: ON
- DC: `DC_V_ctrl_on`: OFF
- DC: `conv_on`: ON
- PM: `PM_spd_ctrl`: OFF
- PM: `PM_FOC`: ON
- PM: `conv_on`: ON

With these button settings, speed control for the brushed DC machine is enabled together with sensored field-oriented torque control of the PMsyn machine. This implies that the following set of users sliders are active:

- `n_ref DC(rpm)`: Speed control of the DC brushed motor speed rpm.
- `Id_ref(A)`: Control of the PM direct axis current. Typically, this setting should be set to zero, as torque is controlled by the quadrature component.
- `Iq_ref(A)`: Control of the PM quadrature axis current, which in turn controls the shaft torque.
- `Kp(mNms/rad)`: proportional gain used by the brushed DC speed controller.
- `Kp(mNm/rad)`: Integral gain used by the brushed DC speed controller.

An example of simulated drive operation, given in figure 8.11 is based on a speed control slider setting of 800 rpm and quadrature slider reference of -1.5 A (the direct axis reference current is held at 0 A). The armature current ramp setting is equal to 50 A/s, hence at start up the armature current increases until the speed DC speed control max limit of 1.0 A is reached at time mark 0.02 s. The error between reference and measured (using the PM encoder signal) shaft speed is such that the current limit of the speed controller is maintained until time mark ≈0.145 s. After said time mark the speed controller will endeavour to maintain the shaft speed at the reference speed. At start-up the PM quadrature current will increase in the negative direction (given a current ramp setting of 50 A/s), which causes a PM machine torque build-up to its steady state value at time mark 0.03 s, as may be observed from figure 8.11, At start-up the drive accelerates because the DC brushed machine torque is greater then the PM load torque. Under steady-state shaft speed conditions the DC torque matches the PM machine torque in amplitude but opposite in polarity. For the conditions shown here, energy flow is from the DC machine to the PM machine, i.e this unit functions as a generator. If for example, the quadrature slider reference is reversed (and maintaining the same speed reference setting) to 1.5 A, the energy flow will be reversed, i.e the DC machine will then act as a generator and the PM machine as a motor. The steady state shaft speed will not be affected by the load change.

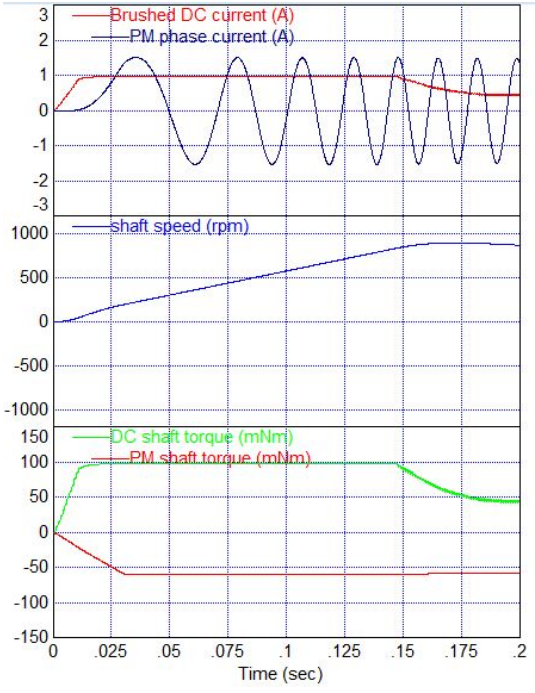


Fig. 8.11 Results: DC-speed/PM torque control: SIM

8.1.1.3 PM-Speed control, DC-Torque control: SIM

Controller button settings:

- DC: `DC_spd_ctrl`: OFF
- DC: `DC_V_ctrl_on`: OFF
- DC: `conv_on`: ON
- PM: `PM_spd_ctrl`: ON
- PM: `PM_FOC`: ON
- PM: `conv_on`: ON

With these button settings, field-oriented speed control for the PMsyn machine is enabled together with torque control of the brushed DC machine. This implies that the following set of users sliders are active:

- `n_ref (rpm)`: Speed control of the PM motor speed rpm.
- `Id_ref(A)`: Control of the PM direct axis current. Typically, this setting should be set to zero, as torque is controlled by the quadrature component.
- `IA_ref(A)`: Control of the brushed DC armature current, which in turn controls the shaft load torque.
- `Kp(mNms/rad)`: proportional gain used by the PMsyn speed controller.
- `Kp(mNm/rad)`: Integral gain used by the PMsyn speed controller.

An example of simulated drive operation, given in figure 8.12 is based on a speed control slider setting of 600 rpm and armature current slider reference of -0.5 A. The direct axis reference current of the PM machine is held at 0 A. At start-up the speed error seen by the PM speed controller is such

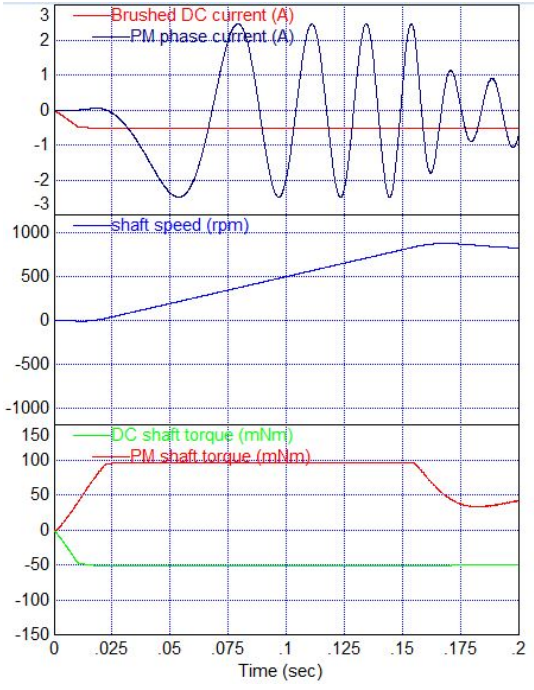


Fig. 8.12 Results: PM-speed/DC torque control: SIM

that the output limit is reached which in this example implies a quadrature current maximum value of 2.5 A. This current limit is reflected in the PM shaft torque, as may be observed from figure 8.12. The load torque is set by the armature current, which ramps up to its steady-state value as defined by the current ramp rate. The drive accelerates given the torque difference between motor/load and at ≈0.16 s the shaft speed reaches the reference speed. From that time mark onwards the PM torque is reduced (by the speed controller) to reach the steady-state condition, where the amplitudes of PM and DC machines are equal but opposite in polarity. Under there conditions, the PM machine acts as a motor and the DC machine as a generator. This energy flow from PM→DC can, for example, be reversed by changing the DC armature current setting from -0.5 A→0.5 A, in which case the DC machine operates under 'motoring' conditions, and the PM in 'generating' mode. The shaft speed, remains at the reference speed setting set by the PM speed controller.

8.1.2 Dual drive sensored DC-PM:HIL

Reference files:

- `DC-PM_0049_GAN-compile.vsm`, for HIL-compile
- `DC-PM_0049_GAN-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) given in figure 8.13 shows the two the `BOOSTXL-GANINV` converters (located on the `LAUNCHXL-F280049` module) which are connected to a PMsyn Teknic machine [2] and brushed DC machine[25]. For HIL operation the drive board

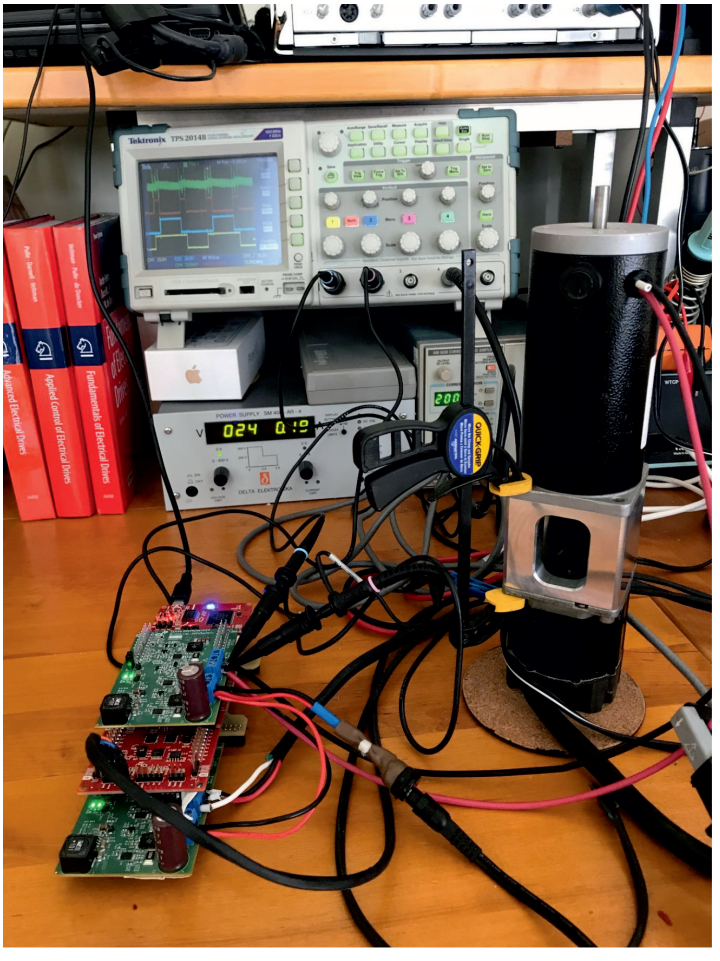


Fig. 8.13 Dual drive setup for DC-PM operation

itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 8.14 shows the 'DC:CC-VC-Sp PM:FOC and C&F' module, which must be compiled to generate an out-file . Inputs, shown in the stage as constants

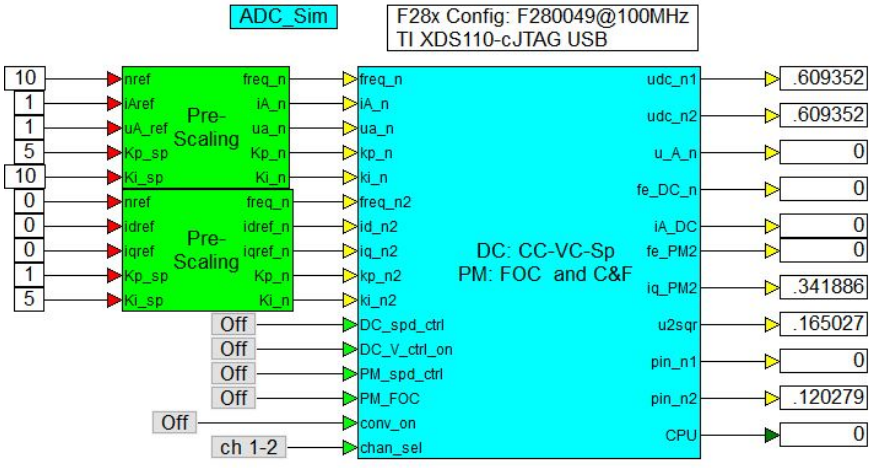


Fig. 8.14 HIL-Compile model of a Dual DC-PMsyn drive

(for readability purposes) for this dual control application are linked to the DC and PM controller modules. Inputs for the DC controller are the per unit: reference frequency `freq_n`, armature current `ia_n`, armature voltage `ua_n` and proportional/integral speed gain inputs `Kp_n`/ `Ki_n`. Per unit inputs for the PM control unit are: reference frequency `freq_n2`, direct/quadrature axis reference current `id_n2`/`iq_n2` and the proportional/integral speed gain inputs `Kp_n2`/ `Ki_n2`.
Six logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other four are used to select the operational mode of the dual drive as discussed earlier (see sections 8.1.1.1, 8.1.1.2 and 8.1.1.3). Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n1` for the brushed DC converter, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.

- The per unit DC bus voltage `udc_n2` for the PMsyn converter, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- The per unit measured brushed DC terminal voltage `u_A_n`.
- The per unit brushed DC frequency `fe_DC_n` used to provide a shaft speed reading.
- The measured per unit armature current `iA_DC_n`, used to generate a shaft torque estimate.
- The per unit PM frequency `fe_PM2` used to provide a shaft speed reading.
- The per unit measured quadrature current `iq_PM2` used to generate a PM shaft torque estimate.
- The per unit measured PM terminal voltage `u2sqr`.
- A per unit estimate of the brushed DC motor input power `pin_n1`.
- A per unit estimate of the PMsyn motor input power `pin_n2`.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 8.14 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 8.15. Four dialogue boxes are used, which contain the parameters that

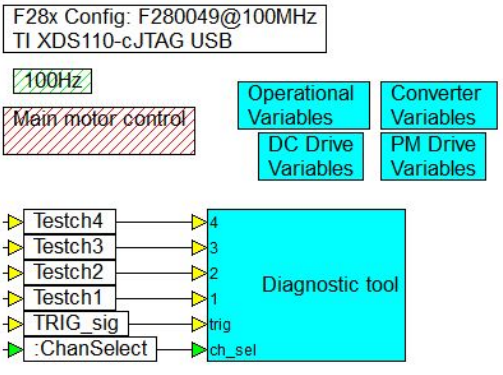


Fig. 8.15 One level down into the Dual Controller module

need to be assigned by the user for HIL-operation. The 'operational Variables' dialogue box, see figure 4.190, is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage/frequency.

In addition, the PMW converter frequency, controller sampling frequency (equal for both PM and DC) are shown. Furthermore, the speed controller sampling frequency is set to 1 kHz. The two dialogue boxes given in figure 8.17

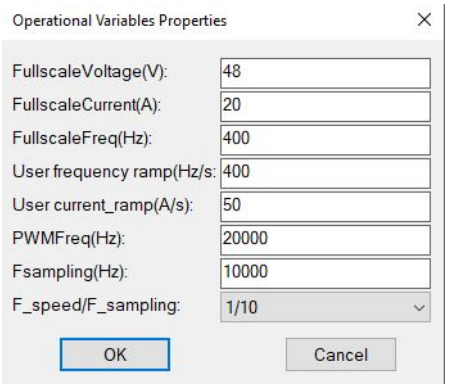
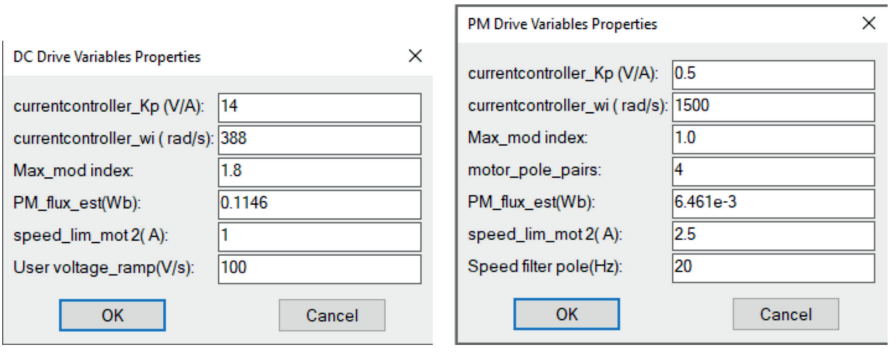


Fig. 8.16 Dialogue box 'Operational Variables'

define the current controller gain/bandwidth for both machines, as well as the maximum modulation indices. For the PM converter this corresponds to a maximum terminal voltage vector amplitude of 12 V and for the brushed DC, the value is set to 21.6 V. The speed control current limits for both machine are also given, which are set to 1 A and 2.5 A for the DC and PM machine respectively. For the DC drive a `User voltage ramp (V/s)` variable



(a) DC Drive variables (b) PM Drive variables

Fig. 8.17 Dialogue boxes, 'DC and PM Drive Variables'

is introduced, which sets the rate of voltage change when armature voltage control is active. For completeness, the 'Converter Variables' dialogue box, is also provided (see figure 8.18) which defines the scaling used by the Analogue

to Digital converter.
This module also shows the low-pass filter corner frequency used by the converter for measuring the half-bridge voltages, see figure 4.12. A 'Diagnostic

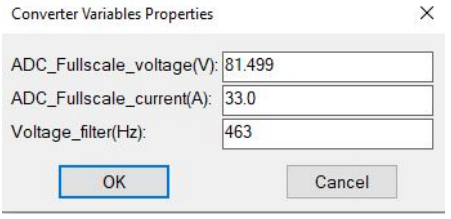


Fig. 8.18 Dialogue box 'Converter Variables'

tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1`, `Test2` or `Test3`, `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (State machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 8.15 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 8.19. Of these, the two controller modules have been discussed earlier, see section 8.1.1. Added to the main controller unit are two `Pow_in` modules used to estimate the input power of the PM and Dc machine. For dual control this is important as it shows the energy flow of the drive.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F280049` with two `BOOSTXL-GANINV` converters

The converter dead-time is set to zero because gate drivers are present, which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module

is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

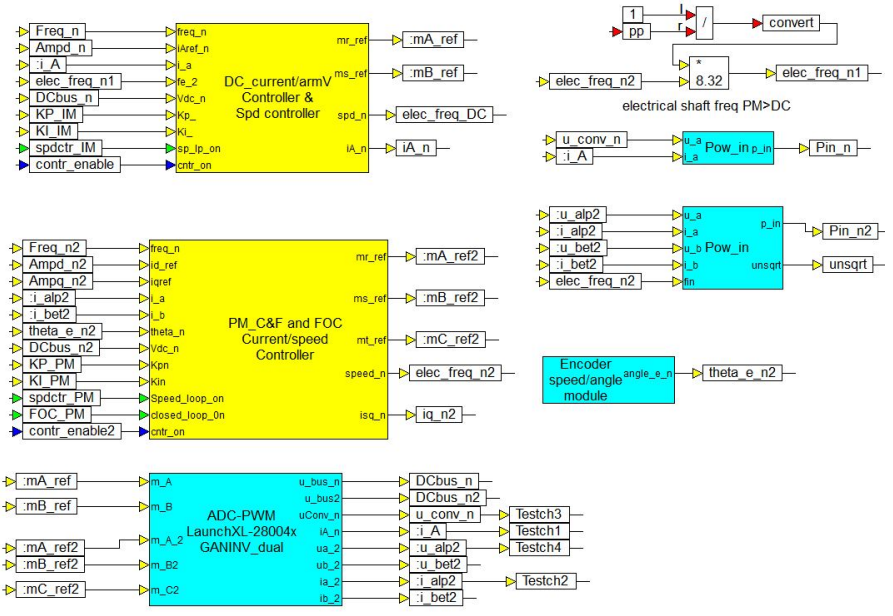


Fig. 8.19 One level down into the 'Main motor controller' module

The `ADC-PWM-LaunchXL-28004x BOOSTXL-GANINV_dual` unit is used to generate the PWM signals for both converters. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages for the PM converter and these are then converted to stationary coordinates α, β voltages: `u_alp2`, `u_bet2` using a forward Clarke conversion. Similarly, the three measured (per unit) PM phase currents are converted to stationary coordinates α, β with outputs `i_alp2`, `i_bet2` respectively. The armature voltage/current outputs `u_con_n/i_A` are derived from the two half-bridges (A and B) used for brushed DC machine control.
ADC Offsets in this unit, are removed with the aid of a low-pass filters, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltages: `DCbus_n`, `DCbus_n2` used to measure the DC bus voltages of both converters, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above). In this case three HIL-debug examples, as mentioned in

the simulation (see section 8.1.1), for three different operating modes will be given in the ensuing subsections.

8.1.2.1 DC-Voltage control, PM-FOC torque control: HIL

The operational HIL example considered here requires the following controller button settings:

- DC: `DC_spd_ctrl`: OFF
- DC: `DC_V_ctrl_on`: ON
- DC: `conv_on`: ON
- PM: `PM_spd_ctrl`: OFF
- PM: `PM_FOC`: ON
- PM: `conv_on`: ON

With these button settings, armature voltage control for the brushed DC machine is enabled together with sensored field-oriented torque control of the PMsyn machine. This implies that the following set of users sliders are active:

- `UA_ref(V)`: Control of the DC brushed motor armature voltage.
- `Id_ref(A)`: Control of the PM direct axis current. Typically, this setting should be set to zero, as torque is controlled by the quadrature component.
- `Iq_ref(A)`: Control of the PM quadrature axis current, which in turn controls the PM shaft torque.

An example of HIL drive operation, given in figure 8.20 is based on an armature voltage slider setting of 10 V and quadrature current slider reference of -1.5 A (the direct axis reference current is held at 0 A). Central to this figure is the Embed 'target execution module' , which executes said out-file. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals. The scope given in figure 8.20 displays the per unit measured armature current ('blue') and PM phase current ('red'). This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit armature voltage and LPF terminal vector voltage component u_α.

Observation of figure 8.20 confirms on the basis of the input power gauge readings that the DC machine is in motoring mode and the PM machine in generating mode. Furthermore the DC machine shaft torque is positive (as is the shaft speed) while the PM shaft torque is negative. Note that these two reading are NOT equal (in absolute value, as shown in the simulation model) due to the presence of a friction torque component of $\approx$25 mNm.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with

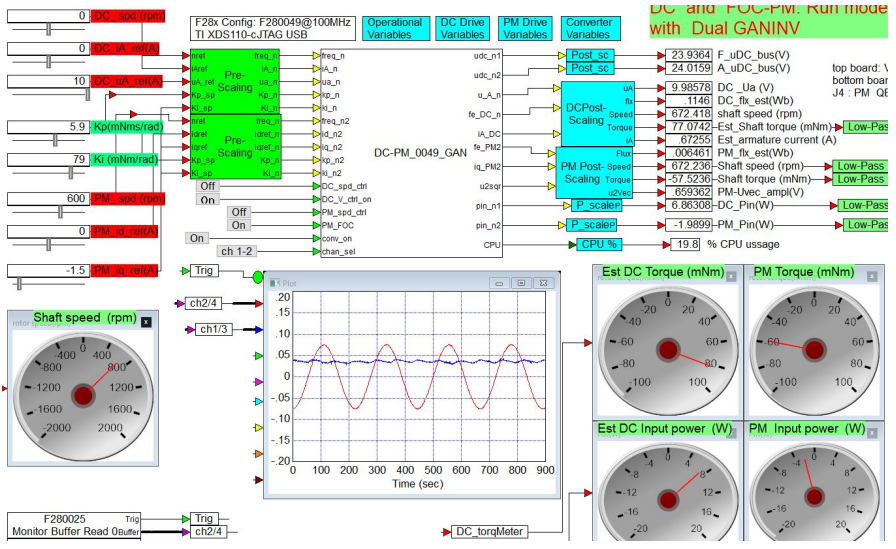


Fig. 8.20 HIL-Debug model, F0049-GANINV of a Dual Brushed DC-PMsyn drive with armature voltage control and FOC torque control

live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

8.1.2.2 DC-Speed control, PM-FOC torque control: HIL

The operational HIL example considered here requires the following controller button settings:

- DC: `DC_spd_ctrl`: ON
- DC: `DC_V_ctrl_on`: OFF
- DC: `conv_on`: ON
- PM: `PM_spd_ctrl`: OFF
- PM: `PM_FOC`: ON
- PM: `conv_on`: ON

With these button settings, armature speed control for the brushed DC machine is enabled together with sensored field-oriented torque control of the PMsyn machine. This implies that the following set of users sliders are active:

- `n_ref DC (rpm)`: Speed control of the DC brushed motor.
- `Id_ref(A)`: Control of the PM direct axis current. Typically, this setting should be set to zero, as torque is controlled by the quadrature component.
- `Iq_ref(A)`: Control of the PM quadrature axis current, which in turn controls the PM shaft torque.
- `Kp(mNms/rad)`: proportional gain used by the brushed DC speed controller.
- `Kp(mNm/rad)`: Integral gain used by the brushed DC speed controller.

An example of HIL drive operation, given in figure 8.21 is based on an speed control slider setting of 800 rpm and quadrature current slider reference of -1.5 A (the direct axis reference current is held at 0 A). Central to this figure is the Embed 'target execution module' , which executes said out-file. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals. The scope given in figure 8.21 displays the per unit measured armature current ('blue') and PM phase current ('red'). This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit armature voltage and LPF terminal vector voltage component u_α.

Observation of figure 8.21 confirms on the basis of the input power gauge readings that the DC machine is in motoring mode and the PM machine in generating mode. Furthermore the DC machine shaft torque is positive (as is the shaft speed) while the PM shaft torque is negative. Note that these two reading are NOT equal (in absolute value, as shown in the simulation example) due to the presence of a friction torque component of $\approx$25 mNm.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to active a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

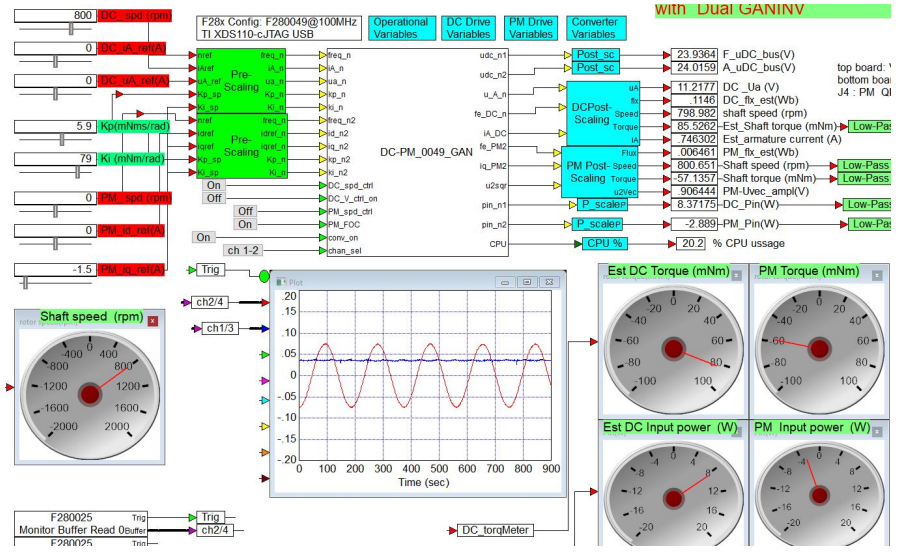


Fig. 8.21 HIL-Debug model, F0049-GANINV of a Dual Brushed DC-PMsyn drive with armature speed control and FOC torque control

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

8.1.2.3 PM-FOC speed control, DC-Torque control: HIL

The operational HIL example considered here requires the following controller button settings:

- DC: `DC_spd_ctrl`: OFF
- DC: `DC_V_ctrl_on`: OFF
- DC: `conv_on`: ON
- PM: `PM_spd_ctrl`: ON
- PM: `PM_FOC`: ON
- PM: `conv_on`: ON

With these button settings, field-oriented speed control for the PMsyn machine is enabled together with sensored torque control of the brushed DC machine. This implies that the following set of users sliders are active:

- `n_ref (rpm)`: Speed control of the PM motor.
- `Id_ref(A)`: Control of the PM direct axis current. Typically, this setting should be set to zero, as torque is controlled by the quadrature component.
- `IA_ref(A)`: Control of the brushed DC armature current, which in turn controls the DC shaft torque.
- `Kp(mNms/rad)`: proportional gain used by the PM speed controller.
- `Kp(mNm/rad)`: Integral gain used by the PM speed controller.

An example of HIL drive operation, given in figure 8.22 is based on an speed control slider setting of 600 rpm and armature current slider reference of -0.5 A. The direct axis reference current of the PM machine is held at 0 A). Central to this figure is the Embed 'target execution module' , which executes said out-file. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals. The scope given in figure 8.22 displays the per unit measured armature current ('blue') and PM phase current ('red'). This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit armature voltage and LPF terminal vector voltage component u_α.

Observation of figure 8.22 confirms on the basis of the input power gauge readings that the DC machine is in generating mode and the PM machine in motoring mode. Furthermore the DC machine shaft torque is negative while the PM shaft torque is positive (as is the shaft speed). Note that these two reading are NOT equal (in absolute value, as shown in the simulation mod) due to the presence of a friction torque component of $\approx$25 mNm.

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

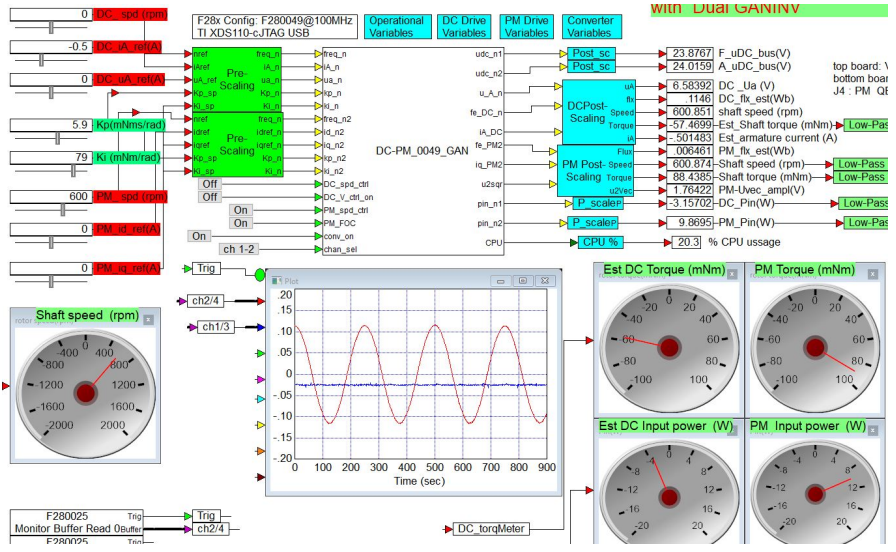


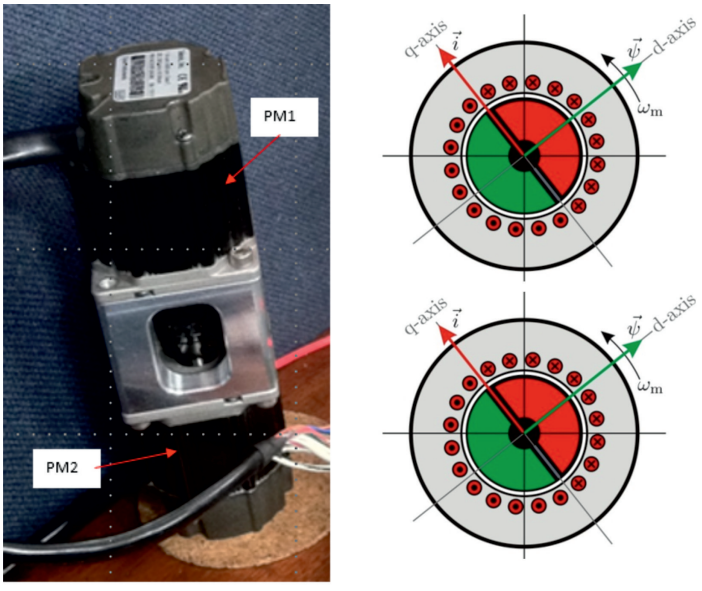
Fig. 8.22 HIL-Debug model, F0049-GANINV of a Dual Brushed DC-PMsyn drive with PM FOC speed control and armature torque control

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

8.2 Dual drive sensored PM-PM

The dual drive considered in this section consists of two mechanically connected PM synchronous machines [2], as shown in figure 8.23. Included in this figure are two models [13] of said machines (identical in this case) , which shows the importance of controlling the current vector relative to the 'dq' coordinate system(s) in order to control the shaft torque. Each PM synchronous machine, has a rotating flux vector, hence a rotating orthogonal (to the flux vector) current vector is required for each machine that is generated in the three-phase distributed winding [28]. The overall drive concept of the PM-PM drive, as



(a) PM-PM Drive (b) Dual PMsyn machines

Fig. 8.23 PM-PM dual drive setup

given in figure 8.24 shows the two controllers that generate the modulation indices for the respective converters that are connected to the machines in question. Both machines are provided with an encoder used to identify the flux vector as required for field-oriented control. An ADC unit is used to provide the currents, LPF converter voltages and encoder signals required for the two controllers. Control for the PM synchronous machines is implemented in 'controller 1', 'controller 2' and typically consists of: sensored field-oriented current and speed control (see section 4.3.1). In addition, open loop current & frequency control (see section 4.2.1) , is usually added to check alignment of the encoder angle with the flux vector.

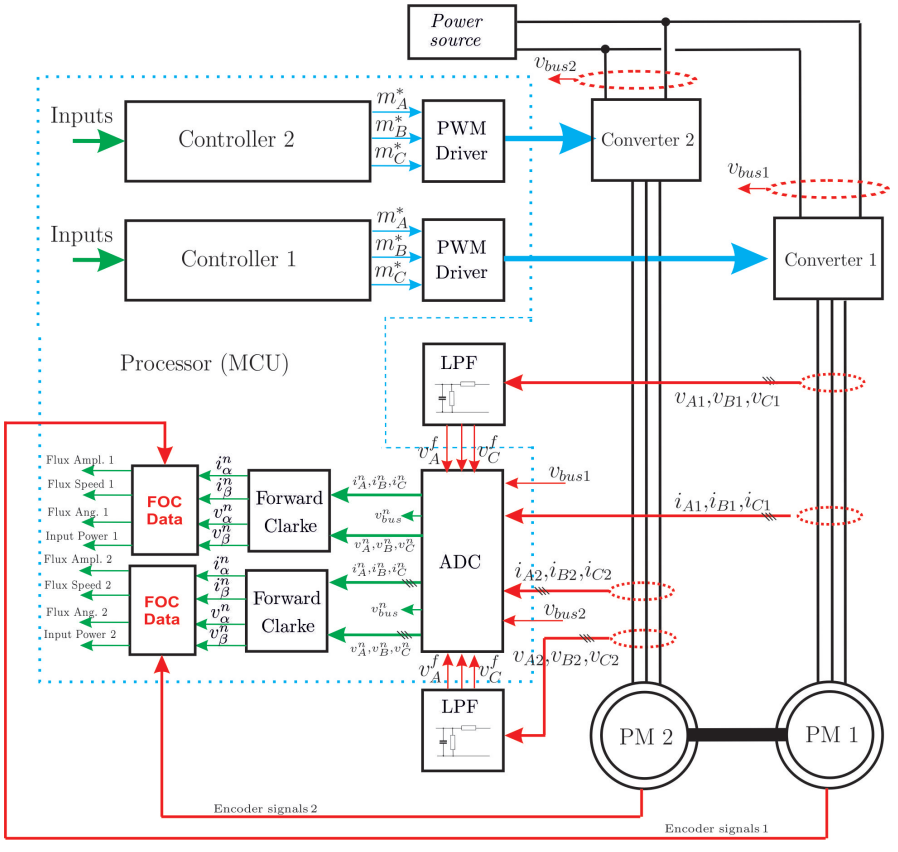


Fig. 8.24 PM-PM dual drive concept

8.2.1 Dual drive sensored PM-PM: SIM

Reference file: `PM-PM_0049_GAN-Sim.vsm`.

This example considers the simulation of the two PMsyn Teknic machines [2] connected to a Texas Instruments F0049 twin GANINV converter setup, as represented in figure 8.25. The aim is to examine operation of the drive using two fixed point controllers, that will subsequently be used in the HIL implementation.

For the simulation example the PM controller `PM2 FOC Torque controller` for machine 2, has been simplified by omitting the speed controller. However, for the HIL example two identical controllers of the type used for machine 1 are used. Furthermore, the shaft speed `nm` and shaft angle `the-me` generated by the `Machine coupling` unit are used to implement field-oriented control for BOTH machines. In the HIL example each machine has its own encoder

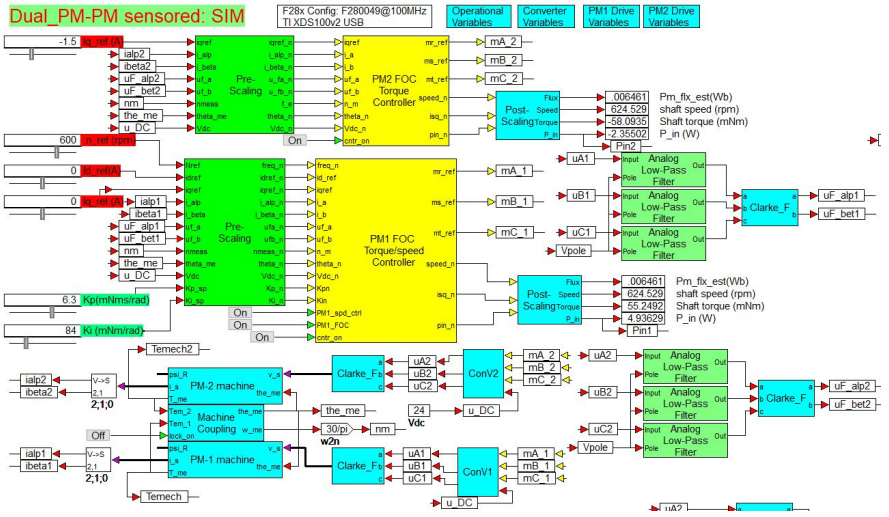


Fig. 8.25 PM-PM dual drive concept: simulation model

(and speed estimator), as will be come apparent shortly.
The IRTF based PM model [13] `PM machine IRTF based` as introduced earlier (see figure 2.14) is connected to an 'average voltage per sample' converter `Conv` that, in turn is linked to the module `current/FOC & Spd controller` which is similar to one introduced earlier (see figure 4.177).
Both machine models are connected to a module `Machine Coupling`, which represents a non-flexible coupling as introduced earlier (for the dual DC-PM example), see expression (8.1).

The PM1 machine controller makes use of three sliders: `n_ref`(PM speed control), `id_ref`(direct axis current control), `iq_ref`(quadrature axis current control), as required for FOC-speed/torque and C&F control.
In addition, two sliders: `K_p`, `K_i` are present, which set the proportional and integral gain values for the PM speed controller. The PM2 machine controller makes use (for the SIM example) of one slider: `iq_ref` (quadrature axis current control) which provides torque control for PM2.

The simulation result given in figure 8.26 for a time sequence of 0.2 s, corresponds to FOC speed control for machine PM1 with a speed reference slider setting of 600 rpm. The quadrature current slider for machine PM2 is to -1.5 A. for both machines the direct axis current is set to zero. At start-up, the PM 1 machine speed controller is at this maximum quadrature current limit of 2.5 A as the measured speed is less then the reference value. At time mark ≈0.13 s the measured speed reaches the reference speed, in which case the speed controller sets the quadrature current reference to meet the (steady-state) condition: shaft torque PM1 is equal but opposite to the torque produced by

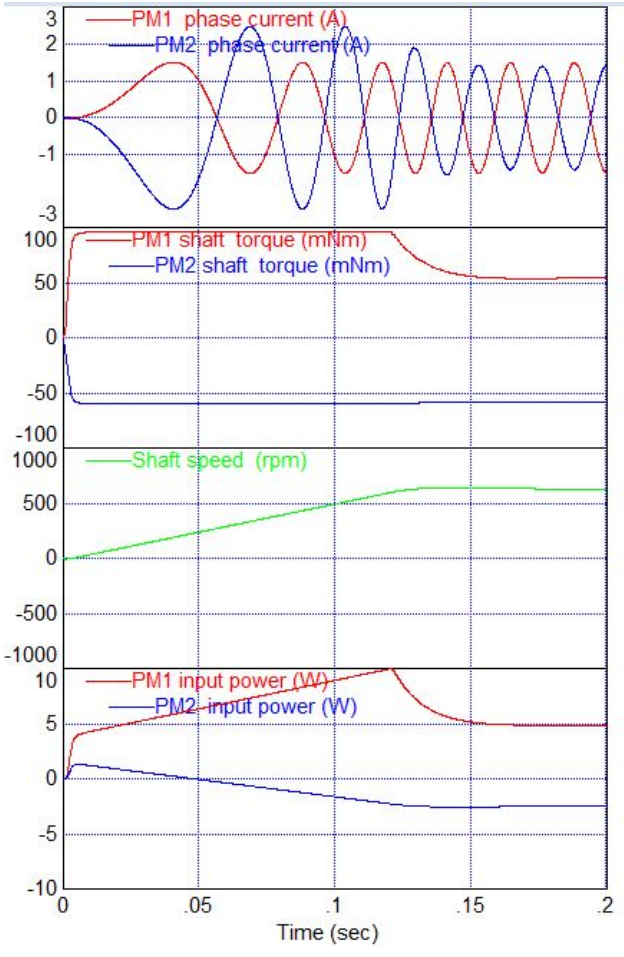
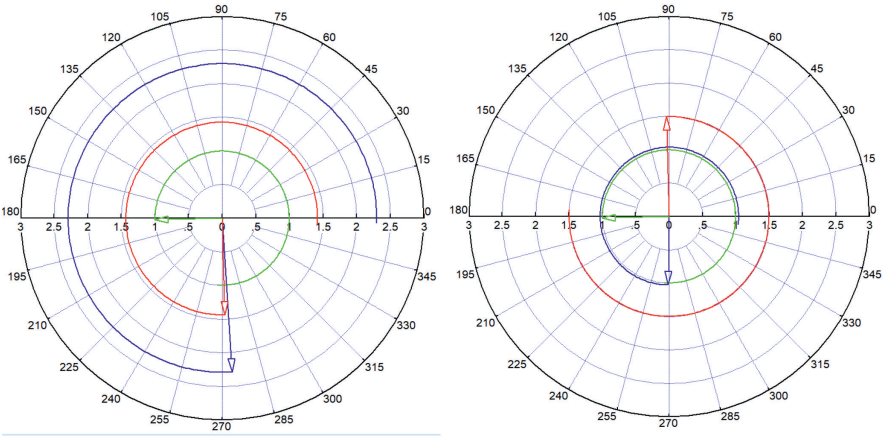


Fig. 8.26 Results: PM-PM dual drive concept, simulation model

PM2.
This condition is also apparent from the two vector plots (see figure 8.27) taken at time mark 0.2 s. Shown in figure 8.27(a) is the unity d-axis ('green'), current vector ('red') and terminal voltage vector ('blue') for machine PM1. The orientation of the current vector,relative to the d-axis (leading by 90 degrees) confirms that said machine is in motoring mode and generates a quadrature e current, which is equal but opposite to the quadrature currents set by the user for machine PM2 (see figure 8.27(b)). This also evident from the phase currents of both machines, as shown in figure 8.26. Note that in the SIM model the 'dq' coordinates are aligned for BOTH machines and that is usually not the case in the HIL (but can be achieved by adjusting the two coordinate frames to achieve alignment as shown in this SIM example). In



(a) PM-1 Drive (b) PM-2 Drive

Fig. 8.27 Results: Vector plots PM-PM dual drive concept, simulation model

this example the (steady-state) energy flow is from machine PM1 to PM2, as may be observed from the input power values at time mark 0.2 s. However, if the quadrature current setting for PM2 is changed from $-1.5 \rightarrow 1.5$A (by the user) the energy flow will be reversed, i.e PM2 will be in motoring mode, PM1 in generating mode and shaft speed will not be affected.

8.2.2 Dual drive sensored PM-PM: HIL

Reference files:

- `PM-PM_0049_GAN-compile.vsm`, for HIL-compile
- `PM-PM_0049_GAN-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) given in figure 8.28 shows the two `BOOSTXL-GANINV` converters (located on the `LAUNCHXL-F280049` module), which are connected to two PMsyn Teknic machines [2]. For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 8.29 shows the 'DC:CC-VC-Sp PM:FOC and C&F'

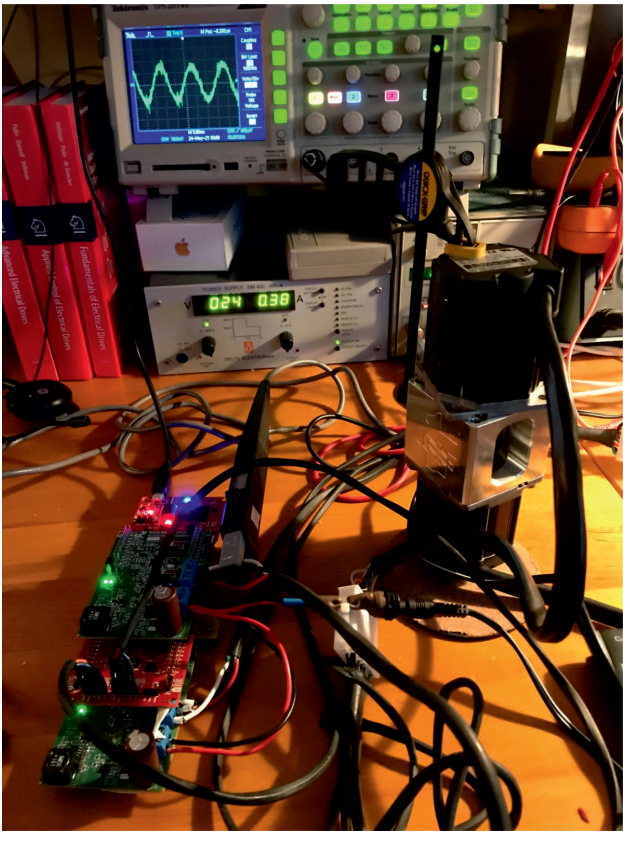


Fig. 8.28 Dual drive setup for PM-PM operation

module, which must be compiled to generate an out-file . Inputs, shown in the stage as constants (for readability purposes) for this dual control application are linked to both PM controller modules which are identical. The per unit inputs for PM1 control unit are: reference frequency `freq_n`, direct/quadrature axis reference current `id_n`/`iq_n` and the proportional/integral speed gain inputs `Kp_n`/ `Ki_n`. An identical set of input variables is also introduced for the PM2 control unit.
Six logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other four are used to select the operational mode of the dual drive as discussed earlier (see section 8.2.1). Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n1` for the top (above the F0049 board) converter, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.

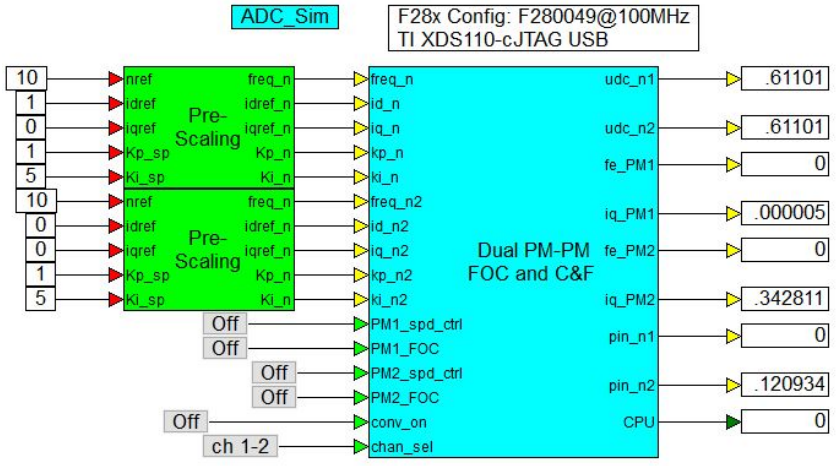


Fig. 8.29 HIL-Compile model of a Dual DC-PMsyn drive

- The per unit DC bus voltage `udc_n2` for the bottom (underneath the F0049 board) converter, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- The per unit frequency `fe_PM1` used to provide a shaft speed reading for machine PM1.
- The per unit measured quadrature current `iq_PM1` used to generate a shaft torque estimate for machine PM1.
- The per unit PM frequency `fe_PM2` used to provide a shaft speed reading for machine PM2.
- The per unit measured quadrature current `iq_PM2` used to generate a shaft torque estimate for machine PM2.
- A per unit estimate of the PM1 machine input power `pin_n1`.
- A per unit estimate of the PM2 machine input power `pin_n2`.
- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 8.14 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 8.30. Four dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'operational Variables' dialogue box, see figure 8.31, is used to assign full scale values for the voltage,

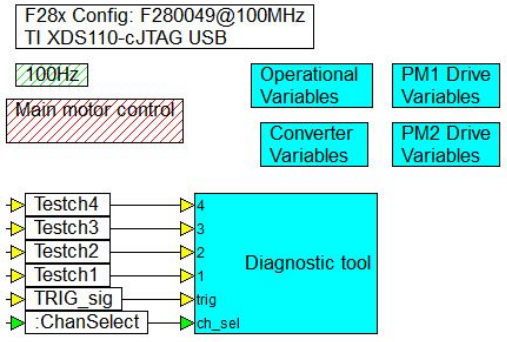


Fig. 8.30 One level down into the Dual Controller module

current, frequency and the ramp change for the reference voltage/frequency. In addition, the PMW converter frequency, controller sampling frequency (equal for both PM converters) are also shown. Furthermore, the speed controller sampling frequency is set to 1 kHz. The dialogue box given in figure 8.32 defines

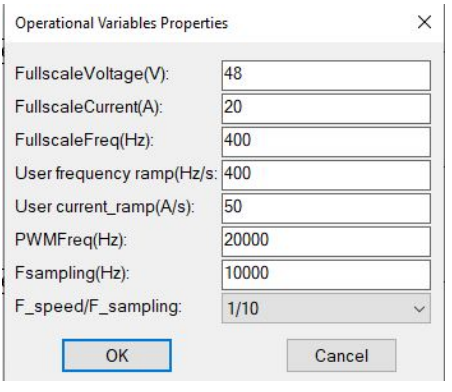


Fig. 8.31 Dialogue box 'Operational Variables'

the current controller gain/bandwidth, as well as the maximum modulation index for machine PM1. This corresponds to a maximum terminal voltage vector amplitude of 12 V. The speed control current limit is also included and is set to 2.5 A for machine PM1. The dialogue box `PM2 Drive Variables` has the same data and is therefore not shown here. The 'Converter Variables' dialogue box, introduced earlier(see figure 8.18) defines the scaling used by the Analogue to Digital converter.
Said dialogue box also shows the low-pass filter corner frequency used by the converter for measuring the half-bridge voltages, see figure 4.12. A 'Diagnostic tool' module is used to buffer two variables, which can subsequently

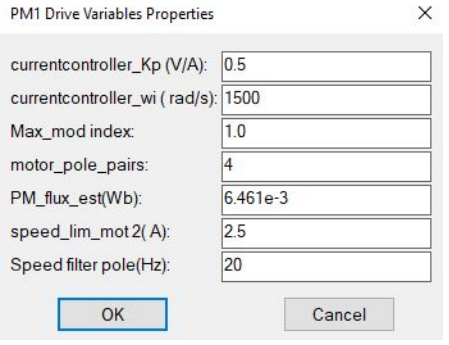


Fig. 8.32 Dialogue box, 'PM1 Drive Variables', which is identical to 'PM2 Drive Variables' (same machines used)

displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1`, `Test2` or `Test3`, `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (State machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 8.30 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 8.33. The two controller modules `FOC/speed Controller_PM` are identical and similar to a unit discussed earlier, see section 4.3.1.1. Added to the main controller unit are two `Pow_in` modules used to estimate the input power of both PM machines. For dual control this is important as it shows the energy flow of the drive.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F280049` with two `BOOSTXL-GANINV` converters

The converter dead-time is set to zero because gate drivers are present, which handle this issue. This ADC-PWM module conforms with MCU timing, as

discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

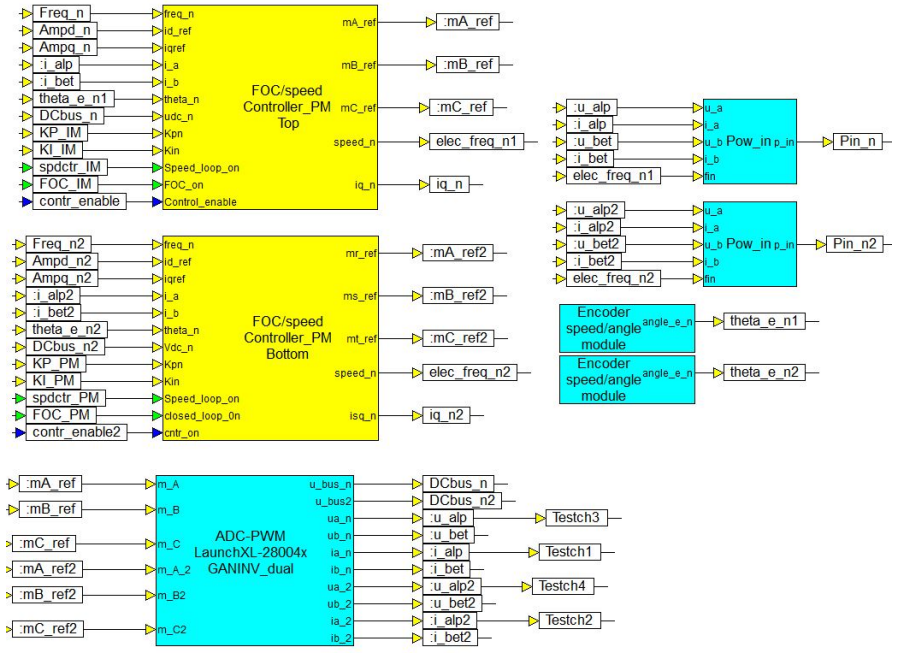


Fig. 8.33 One level down into the 'Main motor controller' module

The `ADC-PWM-LaunchXL-28004x BOOSTXL-GANINV_dual` unit is used to generate the PWM signals for both converters. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages for both PM converters and these are then converted to stationary coordinates α, β voltages: `u_alp2`, `u_bet2` using a forward Clarke conversion. Similarly, the three measured (per unit) PM phase currents are converted to stationary coordinates α, β with outputs `i_alp2`, `i_bet2` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filters, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltages: `DCbus_n`, `DCbus_n2` used to measure the DC bus voltages of both converters, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).

An example of HIL drive operation, given in figure 8.34 is based on an speed reference slider setting of 600 rpm (for PM1) and quadrature current slider reference of -1.5 A (for PM2). The direct axis reference current for both machines is held at 0 A). Central to this figure is the Embed 'target execution module' , which executes said out-file. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals. The scope given in figure 8.34 displays the per unit measured

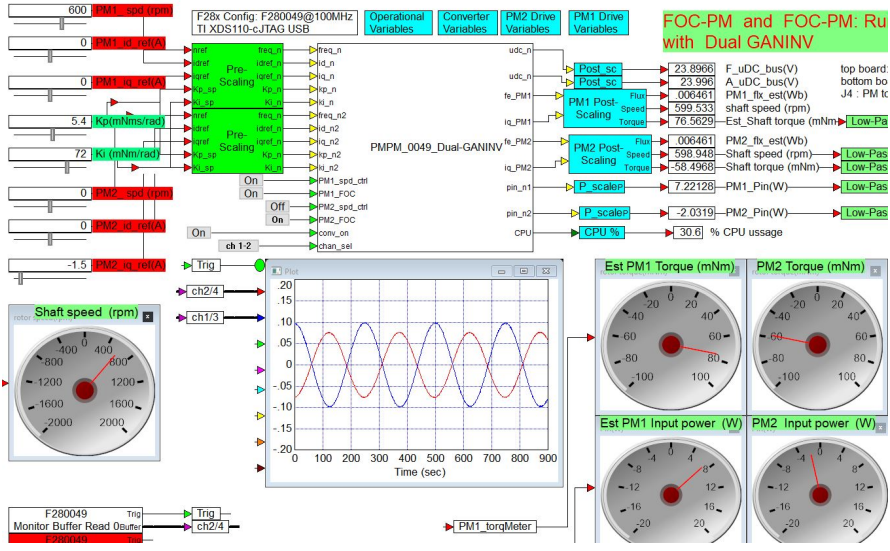


Fig. 8.34 HIL-Debug model, F0049-GANINV of a Dual PMsyn drive with FOC speed control (PM1) and FOC torque control (PM2)

phase current ('blue') of PM1 and phase current ('red') of PM2. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit LPF terminal vector voltage components u_α of both machines. Note that the two phase currents are diametrically opposite as shown earlier in the simulation model (see figure 8.26). However, to achieve a similar result here, the top PM1 motor shown in figure 8.28 was rotated slightly to align the controller d-axis of both machines.

Observation of figure 8.34 confirms on the basis of the input power gauge readings that the PM1 machine is in motoring mode and the PM2 machine in generating mode. Furthermore the PM1 machine shaft torque is positive (as is the shaft speed) while the PM2 shaft torque is negative. Note that these two reading are NOT equal (in absolute value, as shown in the simulation model) due to the presence of a friction torque component of ≈25 mNm.

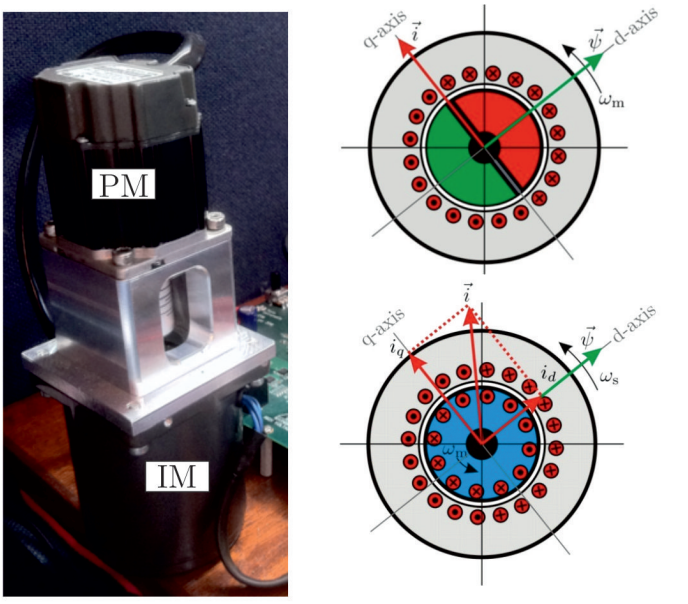
Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

8.3 Dual drive sensored IM-PM

The dual drive considered in this section consists of a three-phase induction machine [25] that is mechanically connected to a PM synchronous machine [2], as shown in figure 8.35. Included in this figure are two models [13] of said machines, which shows the importance of controlling the current vector relative to the 'dq' coordinate system in order to control the shaft torque. The PM synchronous machine, has a rotating flux vector, hence a rotating orthogonal (to the flux vector) current vector is required for said machine that is generated in the three-phase distributed winding [28]. Field-oriented control of the induction machine takes place in the synchronous 'dq' coordinate system and in this case the direct axis component of the current determines the rotor flux amplitude. The quadrature current component in the stator is mirrored in the squirrel cage rotor and this implies the need to generate an EMF e_R (in the rotor) proportional to a slip frequency component (current in the rotor is equal to ${}^{e_R}/_{R_R}$), hence the shaft speed differs from the synchronous speed when load is applied. The overall drive concept of the IM-PM drive as



(a) IM-PM Drive (b) Dual PM IM machines

Fig. 8.35 IM-PM dual drive setup

given in figure 8.36, shows the two controllers that generate the modulation indices for the respective converters that are connected to the machines in question. The PM machine has an encoder used to identify the flux vector as required for field-oriented control and is also used for IM FOC control (which requires a shaft speed estimate). An ADC unit is used to provide the currents, LPF converter voltages and encoder signal required for both two controllers. Control for the PM synchronous machine is implemented in 'controller 2' and typically consists of: sensored field-oriented current and speed control (see section 4.3.1). In addition, open loop current & frequency control (see section 4.2.1) , is usually added to check alignment of the encoder angle with the flux vector. The induction machine is connected to 'Controller 1' and makes use of the encoder connected to the PM machine in order to generate a shaft speed estimate. Modes of operation for the induction machine drive are: field-oriented control and open loop current & frequency control (see section 6.2).

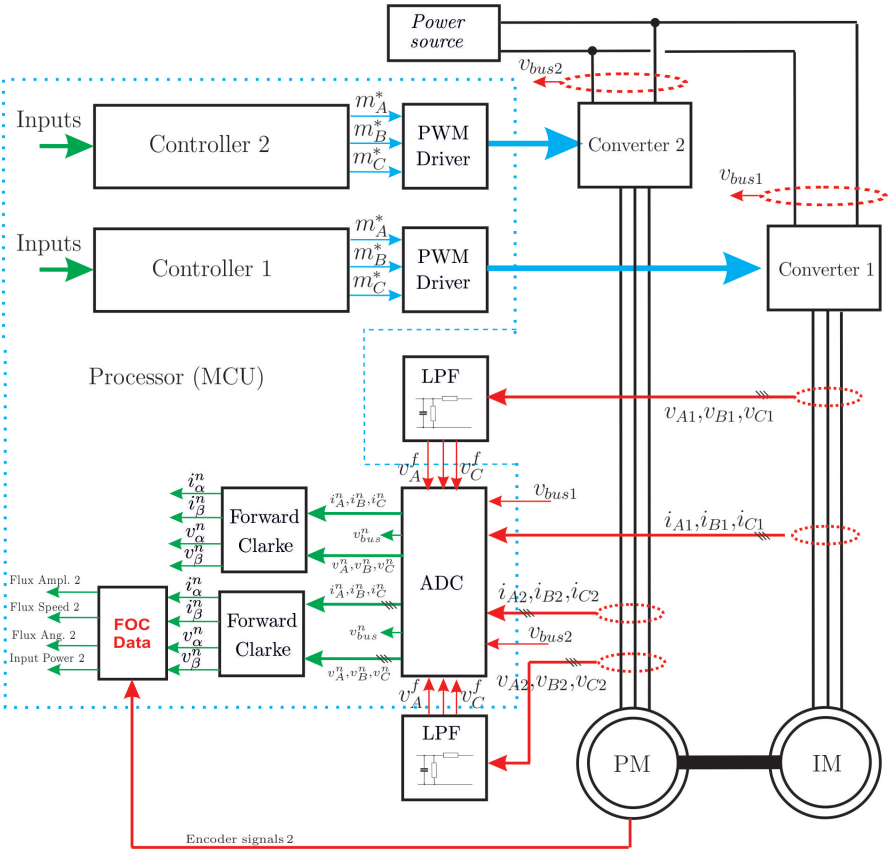


Fig. 8.36 IM-PM dual drive concept

8.3.1 Dual drive sensored IM-PM: SIM

Reference file: `IM-PM_0049C_GAN-SIM.vsm`.

This example considers the simulation of the dual PMsyn Teknic machine [2]-IM machine [25] connected to a Texas Instruments F0049 twin GANINV converter setup, as represented in figure 8.37. The aim is to examine operation (for different operating modes) of the drive using two fixed point controllers, that will subsequently be used in the HIL implementation.

Shown in figure 8.37 are the two controllers `Current/speed controller_PM`, `Current/speed controller_IM` as introduced in sections 4.3.1.1 and 6.2.1.1 respectively. Both are capable of open loop Current & frequency control and field-oriented control, pending the operation mode selected. Note that if FOC is NOT selected for either PM (button `PM_FOC` OFF) or IM (button `IM_FOC` OFF), the variable `isq_n` used to estimate the shaft torque is purposely

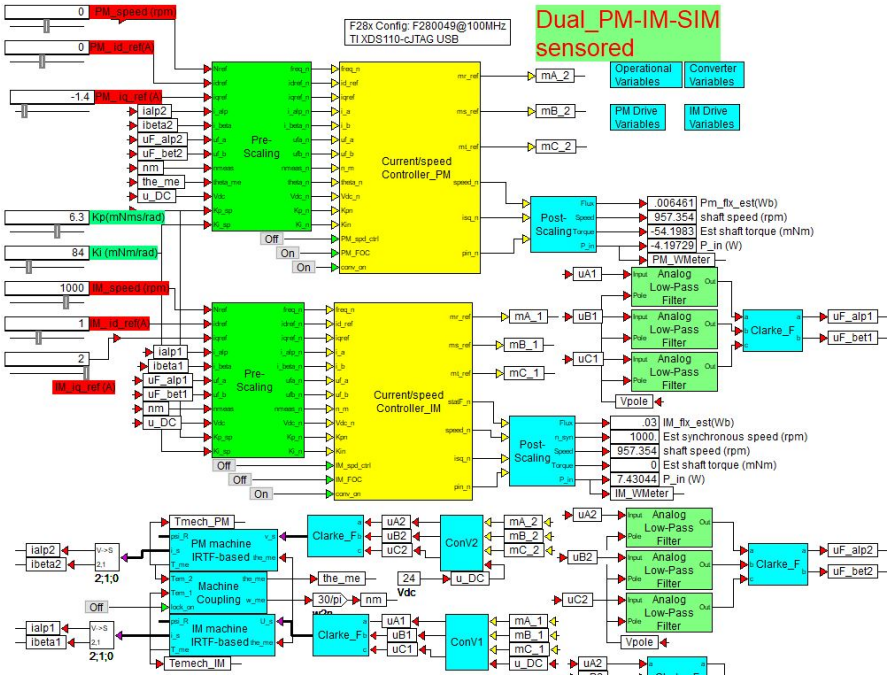


Fig. 8.37 IM-PM dual drive concept: simulation model

turned off (set to zero to avoid confusion). The IRTF based PM model [13] `PM machine IRTF based` as introduced earlier (see figure 2.14) is connected to an 'average voltage per sample' converter `Conv2` that, in turn is linked to the module `current/speed controller_PM`. An IRTF based IM model [13] `IM machine IRTF based` as introduced earlier (see figure 2.25) is connected to an 'average voltage per sample' converter `Conv1` that, in turn is linked to the module `current/speed controller_IM`.
Both machine models are connected to a module `Machine Coupling` which represent a non-flexible coupling as introduced earlier (see expression (8.1)).

For IM machine control three sliders: `IM_speed`(IM speed control), `IM_id_ref` (direct axis current control, which sets the rotor flux if FOC is selected), `IM_iq_ref`(quadrature axis current control) are available, pending the operation mode selected, as will be discussed shortly. The PM machine controller also makes use of three sliders: `PM_speed`(PM speed control), `PM_id_ref`(direct axis current control), `PM_iq_ref`(quadrature axis current control), as required for FOC-speed/torque control.
In addition, two sliders: `K_p`, `K_i` are present, which set the proportional and integral gain values for both IM/PM speed controllers (of which only one is used at any time).

A total of six buttons are connected to these controllers, which define the operation modes of the dual drive. In the ensuing subsections, relevant operation modes are discussed together with the results obtained.

8.3.1.1 IM-Open loop C&F control, PM-FOC torque control: SIM

The operational model considered here requires the following controller button settings:

- IM: `IM_spd_ctrl`: OFF
- IM: `IM_FOC`: OFF
- IM: `conv_on`: ON
- PM: `PM_spd_ctrl`: OFF
- PM: `PM_FOC`: ON
- PM: `conv_on`: ON

With these button settings, open-loop current & frequency control for the induction machine is enabled together with sensored field-oriented torque control of the PMsyn machine. This implies that the following set of users sliders are active:

- `IM_speed(rpm)`: Sets the synchronous shaft speed of the IM drive.
- `IM_id_ref(A)`: Control of the IM direct axis current.
- `IM_iq_ref(A)`: Control of the IM quadrature axis current.
- `PM_id_ref(A)`: Control of the PM direct axis current. Typically set to zero in FOC mode (as is the case here)
- `PM_iq_ref(A)`: Control of the PM quadrature axis current, which controls the shaft torque when in FOC mode (as is the case here).

Note that in this example the induction machine is controlled by a rotating (speed set by the IM speed slider) current vector defined as $\vec{i} = i_d + ji_q$. An example of simulated drive operation, given in figure 8.38 is based on a IM speed slider setting of 1000 rpm, IM quadrature slider reference of 2 A and IM direct axis reference current 1 A. For the PM drive, the direct axis current ref is held at zero, while the quadrature current reference is set to -1.4 A. The PM current ramp setting is equal to 100 A/s, hence at start up the load torque increases until its steady-state value is reached at time mark 0.014 s, as may be observed from figure 8.9. At start-up the drive accelerates (towards the synchronous speed) because the IM machine torque is marginally greater then the PM load torque. The steady-state shaft speed is reached when the IM torque matches the PM machine torque in amplitude but opposite in polarity. For the conditions shown here, energy flow is from the IM machine to the PM machine, i.e this unit functions as a generator. If for example, the PM quadrature slider reference is reversed (and maintaining the same direct/quadrature current and speed settings for the IM) to 1.4 A, the energy

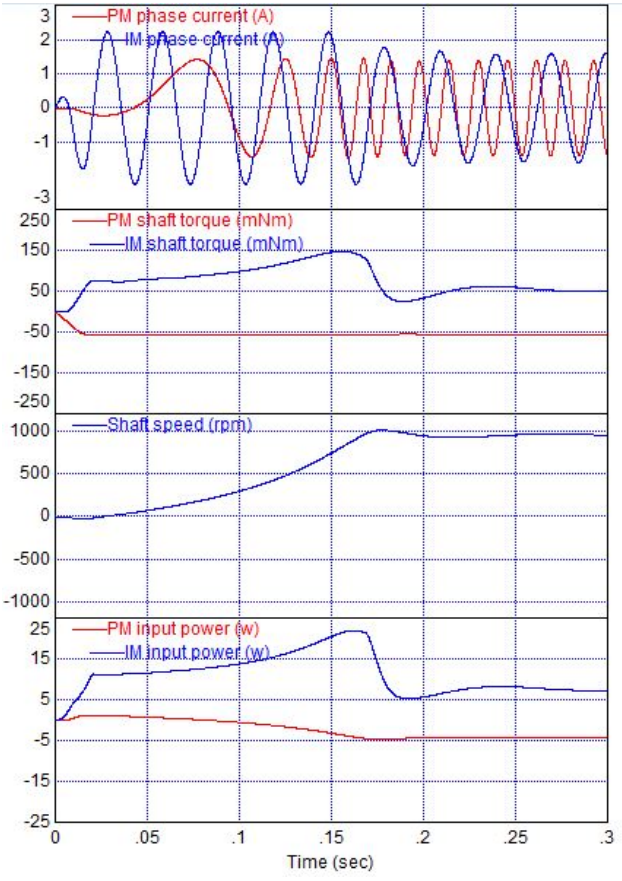
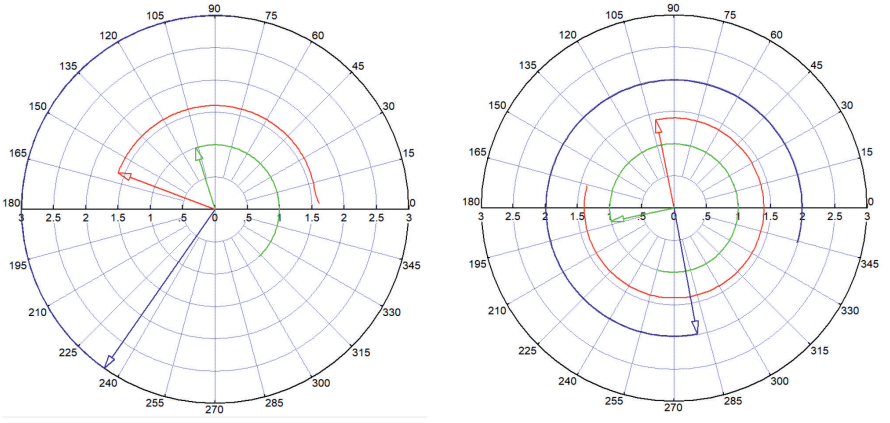


Fig. 8.38 Results: IM-C&F/PM-FOC-torque control: SIM

flow will be reversed, i.e the IM machine will then act as a generator and the PM machine as a motor. The shaft speed will then increase, i.e greater then the set synchronous speed (as expected because the IM is operating in the generating (plugging) mode).
The vector plot given in figure 8.39 corresponds to time mark 0.3 s for the simulation shown in figure 8.38 and the colour legend for vectors shown are: terminal voltage vector ('blue' and IM vector scaled by a factor 1/4), unity direct axis ('green') and current vector ('red'). Figure 8.39(b)) confirms that the PM machine is in generating mode (quadrature current component lagging the d-axis which is tied to the PM flux). Also given is the IM vector plot (figure 8.39(a)), which shows the orientation of the current vector relative to the IM d-axis. Note that the location of the IM rotor flux vector depends on the load applied. With the present load, said rotor flux vector lags the current

vector by ≈15 deg. Note that the vectors are rotating counter clockwise, e.g with positive shaft speed.



(a) IM vector plot (b) PM vector plot

Fig. 8.39 Results: IM-C&F/PM-FOC-torque control, vector plot: SIM

8.3.1.2 IM-FOC speed control, PM-FOC torque control: SIM

The operational model considered here requires the following controller button settings:

- IM: `IM_spd_ctrl`: ON
- IM: `IM_FOC`: ON
- IM: `conv_on`: ON
- PM: `PM_spd_ctrl`: OFF
- PM: `PM_FOC`: ON
- PM: `conv_on`: ON

With these button settings, field-oriented speed control for the induction machine is enabled together with sensored field-oriented torque control of the PMsyn machine. This implies that the following set of users sliders are active:

- `IM_speed(rpm)`: Sets the shaft speed of the IM drive.
- `IM_id_ref(A)`: Control of the IM direct axis current and (with FOC active) the rotor flux amplitude.
- `PM_id_ref(A)`: Control of the PM direct axis current. Typically set to zero in FOC mode (as is the case here).
- `PM_iq_ref(A)`: Control of the PM quadrature axis current, which controls the shaft torque when in FOC mode (as is the case here).

An example of simulated drive operation, given in figure 8.40 is based on a IM speed slider setting of 800 rpm and IM direct axis reference current 1 A, which corresponds to a rotor flux of 30 mWb. For the PM drive, the direct axis current ref is held at zero, while the quadrature current reference is set to -1.4 A.

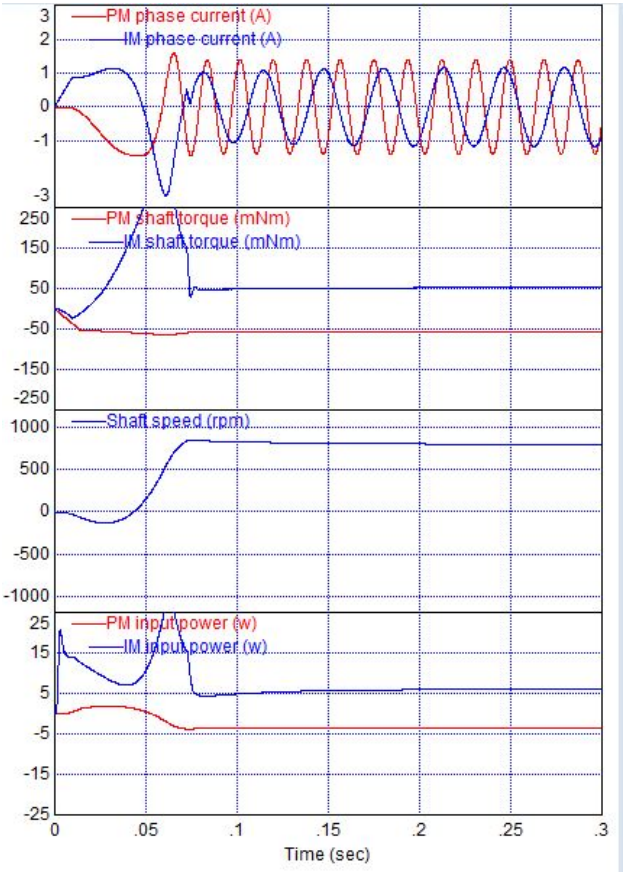
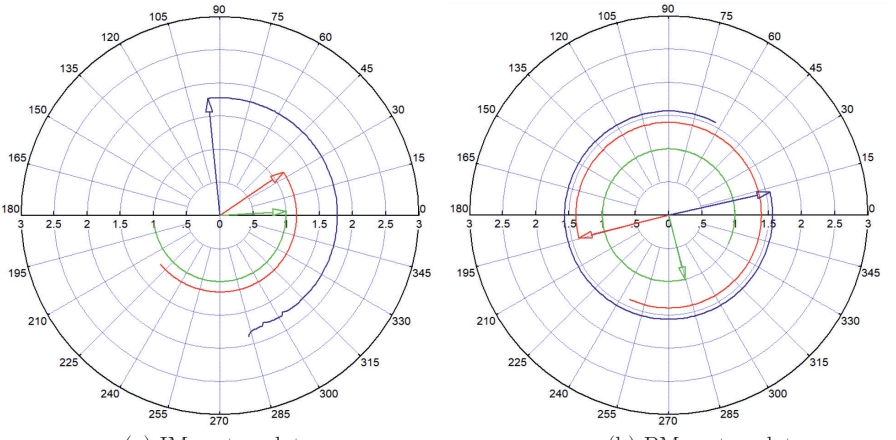


Fig. 8.40 Results: IM-FOC speed/PM-FOC torque control: SIM

The PM current ramp setting is equal to 100 A/s, hence at start-up the load torque increases until its steady-state value is reached at time mark 0.014 s, as may be observed from figure 8.40. At start-up the drive accelerates (towards the reference shaft speed) as the IM machine torque is greater then the PM load torque. The steady-state shaft speed is reached when the IM torque matches the PM machine torque in amplitude but opposite in polarity. Note that start-up of the induction machine is compromised by the fact that rotor flux build needs to take place at the same time as a torque demand is made.

Its recommended, to initially wait for the application of IM torque until the rotor flux is at its steady-state level of 30 mWb in this case. The time constant linked to the flux build up is $\tau_R = {}^{L_M}/_{R_R}$, which is equal to 30 ms for this machine. For the conditions shown here, the steady-state energy flow is from the IM machine to PM machine, i.e this unit functions as a generator. If for example, the PM quadrature slider reference is reversed (and maintaining the same direct/quadrature current and speed settings for the IM) to 1.4 A, the energy flow will be reversed, i.e the IM machine will then act as a generator and the PM machine as a motor. The shaft speed will remain at the reference slider setting.
The vector plot given in figure 8.41 corresponds to time mark 0.3 s for the simulation shown in figure 8.38 and the colour legend for vectors shown are: terminal voltage vector ('blue' and IM vector scaled by a factor 1/4), unity direct axis ('green') and current vector ('red'). Figure 8.41(b)) confirms that the PM machine is in generating mode (quadrature current component lagging the d-axis which is tied to the PM flux). Also given is the IM vector plot (figure 8.41(a)) which shows the orientation of the current vector relative to the IM d-axis. Note that the IM rotor flux vector is now aligned with the direct axis current. Note also that the vectors are rotating counter clockwise, e.g with positive shaft speed.



(a) IM vector plot (b) PM vector plot

Fig. 8.41 Results: IM-FOC speed/PM-FOC torque control, vector plot: SIM

8.3.1.3 IM-FOC torque control, PM-FOC speed control: SIM

The operational model considered here requires the following controller button settings:

- IM: `IM_spd_ctrl`: OFF
- IM: `IM_FOC`: ON
- IM: `conv_on`: ON
- PM: `PM_spd_ctrl`: ON
- PM: `PM_FOC`: ON
- PM: `conv_on`: ON

With these button settings, field-oriented torque control for the induction machine is enabled together with sensored field-oriented speed control of the PMsyn machine. This implies that the following set of users sliders are active:

- `PM_speed(rpm)`: Sets the shaft speed of the PM drive.
- `IM_id_ref(A)`: Control of the IM direct axis current and (with FOC active) the rotor flux amplitude.
- `IM_iq_ref(A)`: Control of the IM quadrature axis current, which controls the IM shaft torque when in FOC mode (as is the case here).
- `PM_id_ref(A)`: Control of the PM direct axis current. Typically set to zero in FOC mode (as is the case here)

An example of simulated drive operation, given in figure 8.42 is based on a PM speed slider setting of 1000 rpm, IM direct axis reference current 1 A, which corresponds to a rotor flux of 30 mWb. The quadrature current reference for the IM is set to -0.6 A. The PM speed current limit is set to 2.5 A, hence at start up the PM shaft torque increases until its speed controller limit value is reached at time mark 0.014 s, as may be observed from figure 8.42. At start-up the drive accelerates (towards the reference shaft speed) as the PM machine torque is greater then the IM load torque. The steady-state shaft speed is reached when the PM torque matches the IM machine torque in amplitude but opposite in polarity. Note that start-up of the induction machine is compromised by the fact that rotor flux build needs to take place at the same time as a torque demand is made.
For the conditions shown here, energy flow is from the PM machine to the IM machine, i.e this unit functions as a generator referred to as the 'plugging mode'. If for example, the IM quadrature slider reference is reversed (and maintaining the same settings for the PM) to 0.6 A, the energy flow will be reversed, i.e the IM machine will then act as a motor and the PM machine moves to generating mode. The shaft speed will remain at the reference slider setting.
The vector plot given in figure 8.43 corresponds to time mark 0.3 s for the simulation time-plot shown in figure 8.42 and the colour legend for vectors shown are: terminal voltage vector ('blue' and IM vector scaled by a factor 1/4), unity direct axis ('green') and current vector ('red'). Figure 8.43(b)) confirms that the PM machine is in motoring mode (quadrature current component leading the d-axis which is tied to the PM flux). Also given is the IM vector plot (figure 8.43(a)) which shows the orientation of the current vector relative to the IM d-axis. Note that the IM rotor flux vector is now aligned with the direct axis current and the quadrature IM current is lagging

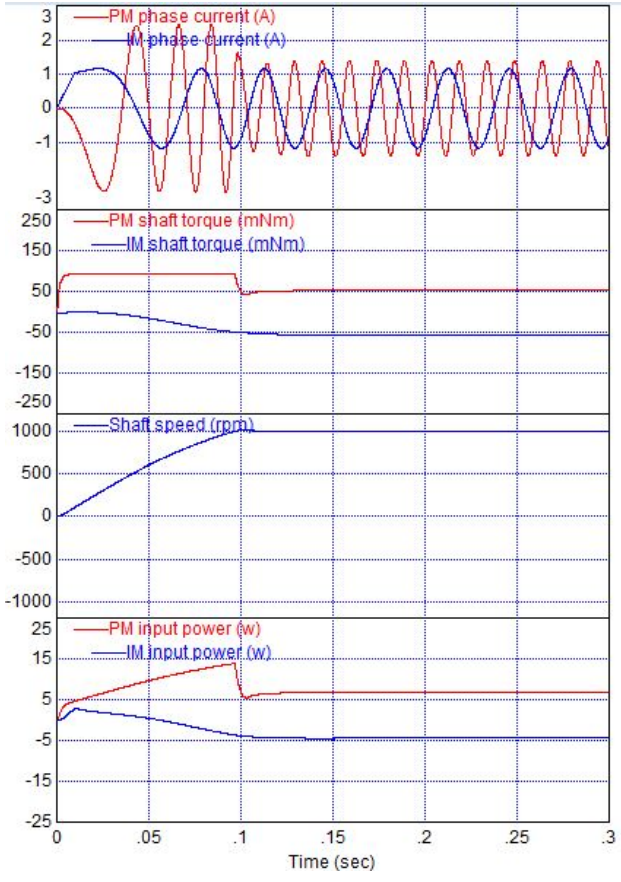
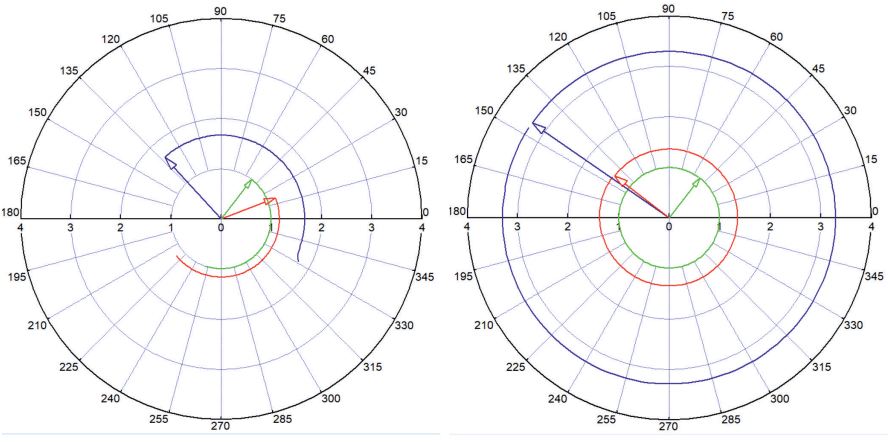


Fig. 8.42 Results: IM-FOC torque/PM-FOC speed control: SIM

the rotor flux vector. Note also that the vectors are rotating counter clockwise, e.g with positive shaft speed.



(a) IM vector plot (b) PM vector plot

Fig. 8.43 Results: IM-FOC torque/PM-FOC speed control, vector plot: SIM

8.3.2 Dual drive sensored IM-PM: HIL

Reference files:

- `IM-PM_0049_GAN-compile.vsm`, for HIL-compile
- `IM-PM_0049_GAN-run.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) given in figure 8.44 shows the two `BOOSTXL-GANINV` converters (located on the `LAUNCHXL-F280049` module), which are connected to a PMsyn Teknic machine [2] and induction machine[25]. For HIL operation the drive board itself needs to be known in

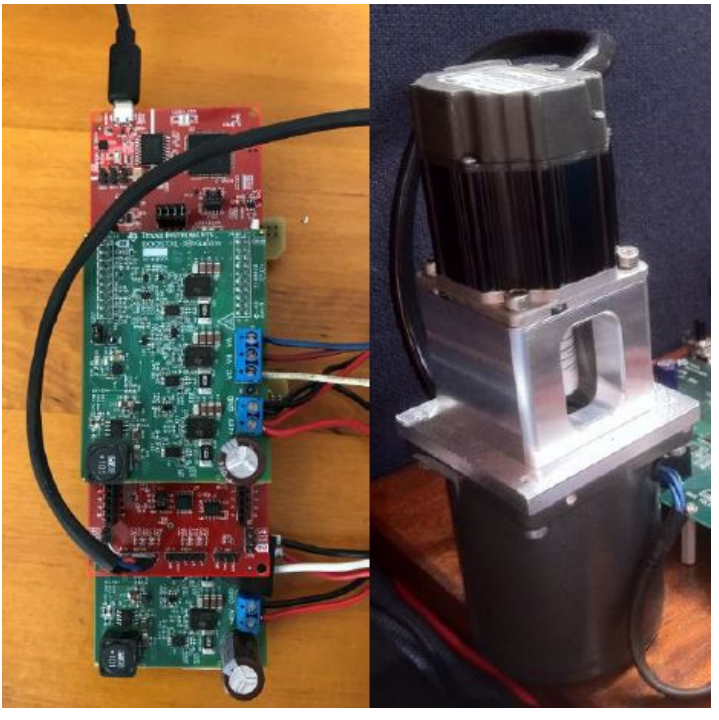


Fig. 8.44 Dual drive setup for IM-PM operation

considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 8.45 shows the

`FOC and CandF IM FOC and CandF PM` module, which must be compiled to generate an out-file . Inputs, shown in this stage as constants (for readability

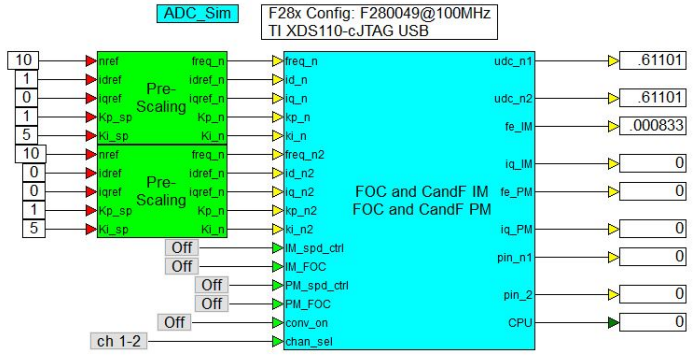


Fig. 8.45 HIL-Compile model of a Dual IM-PMsyn drive

purposes) for this dual control application are linked to the IM and PM controller modules. Inputs for the IM controller are the per unit: reference frequency `freq_n`, direct/quadrature axis reference current `id_n`/`iq_n` and proportional/integral speed gain inputs `Kp_n`/ `Ki_n`. Per unit inputs for the PM control unit are: reference frequency `freq_n2`, direct/quadrature axis reference current `id_n2`/`iq_n2` and the proportional/integral speed gain inputs `Kp_n2`/ `Ki_n2`.
Six logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other four are used to select the operational mode of the dual drive as discussed earlier (see sections 8.3.1.1, 8.3.1.2 and 8.3.1.3). Outputs of the Controller module are:

- The per unit DC bus voltage `udc_n1` for the IM converter, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- The per unit DC bus voltage `udc_n2` for the PMsyn converter, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- The per unit synchronous shaft speed of the IM machine `fe_IM`.
- The measured per unit IM quadrature current `iq_IM`, used to generate a shaft torque estimate. Set to zero if FOC is NOT active.
- The per unit PM frequency `fe_PM` used to provide a shaft speed reading.
- The per unit measured quadrature current `iq_PM` used to generate a PM shaft torque estimate. Set to zero if FOC is NOT active.
- A per unit estimate of the IM motor input power `pin_n1`.
- A per unit estimate of the PMsyn motor input power `pin_n2`.

- A `CPU` output, which provides the user with a % reading of MCU activity, relative to the controller sample time.

Also shown in figure 8.45 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 8.46. Four dialogue boxes are used, which contain the parameters that

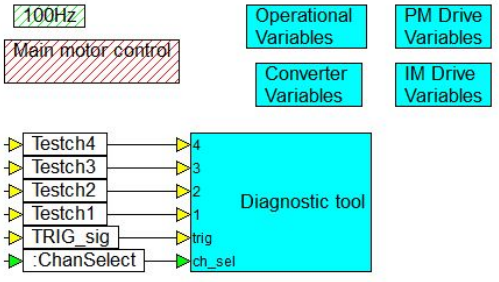


Fig. 8.46 One level down into the Dual Controller module

need to be assigned by the user for HIL-operation. The 'operational Variables' dialogue box, see figure 8.47, is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference voltage/frequency. In addition, the PMW converter frequency, controller sampling frequency (equal for both PM and IM) are shown. Furthermore, the speed controller sampling frequency is set to 1 kHz.
The two dialogue boxes given in figure 8.48 define the current controller gain/bandwidth for both machines, as well as the maximum modulation indices. For the IM converter this corresponds to a maximum terminal voltage vector amplitude of 12 V and for the PM unit , the value is set to 9.6 V. The speed control current limits for both machine are also given, which are set to 1.5 A and 2.5 A for the IM and PM machine respectively. Included in the PM dialogue box is the variable `Speed filter pole (Hz)`, which is the LPF used in the speed estimator (a variable also used for the induction machine drive). The 'Converter Variables' dialogue box, also shown in figure 8.46 was discussed earlier, see figure 8.18.

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables

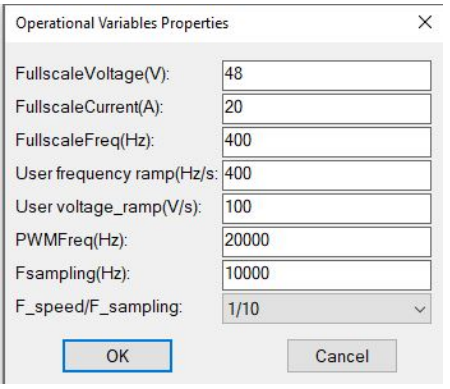
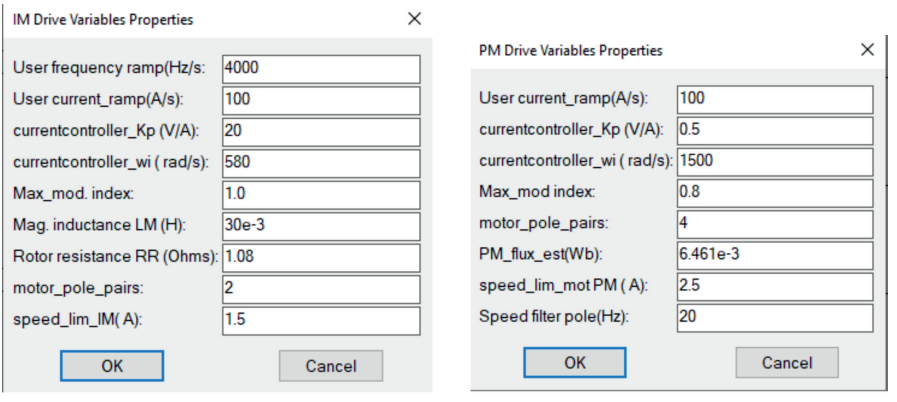


Fig. 8.47 Dialogue box 'Operational Variables'



(a) IM Drive variables (b) PM Drive variables

Fig. 8.48 Dialogue boxes, 'IM and PM Drive Variables'

`Test1`, `Test2` or `Test3`, `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (State machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active.

The key component shown in figure 8.46 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 8.49. Of these, the two controller modules have been discussed earlier,

see sections 4.3.1.4 and 6.2.1.4. Added to the main controller unit are two `Pow_in` modules used to estimate the input power of the PM and IM machine. For dual machine control this is important as it shows the energy flow of the drive.
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 20 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs
- Board in use: `LAUNCHXL-F280049` with two `BOOSTXL-GANINV` converters

The converter dead-time is set to zero because gate drivers are present, which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.

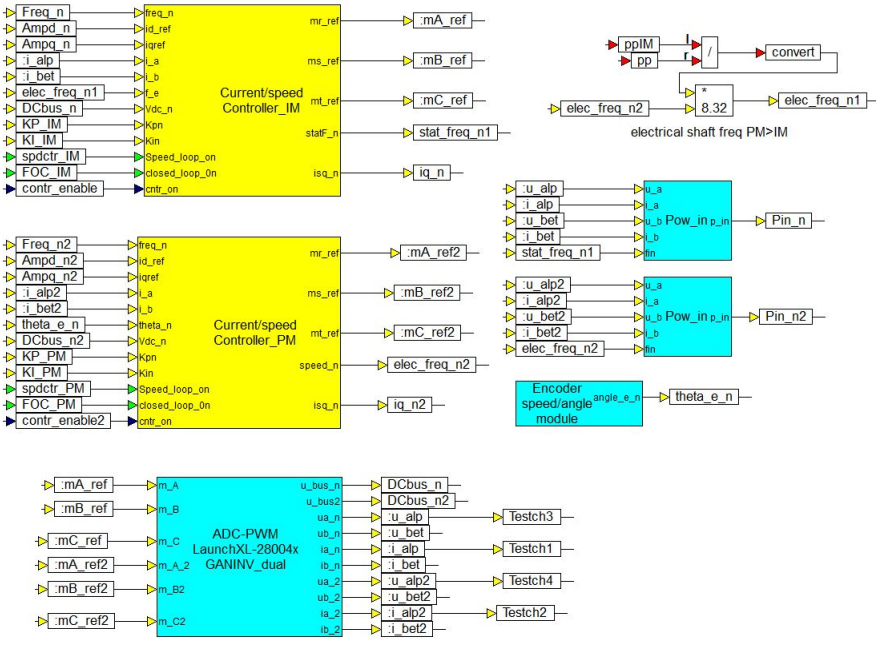


Fig. 8.49 One level down into the 'Main motor controller' module

The `ADC-PWM-LaunchXL-28004x BOOSTXL-GANINV_dual` unit is used to generate the PWM signals for both converters. In addition, it is used to measure the three low-pass filtered (per unit) half-bridge voltages for the PM converter

and these are then converted to stationary coordinates α, β voltages: `u_alp2`, `u_bet2` using a forward Clarke conversion. Similarly, the three measured (per unit) PM phase currents are converted to stationary coordinates α, β with outputs `i_alp2`, `i_bet2` respectively. A similar transformation is also carried for the IM voltages/currents with outputs `u_alp`, `u_bet` and `i_alp`, `i_bet` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filters, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the dialogue box of the ADC/PWM module). Estimation of the ADC offset's is controlled by the '100Hz' module, as mentioned previously. The per unit DC bus voltages: `DCbus_n`, `DCbus_n2` used to measure the DC bus voltages of both converters, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above). In this case three HIL-debug examples, as mentioned in the simulation (see section 8.3.1), for three different operating modes will be given in the ensuing subsections.

8.3.2.1 IM-Open loop C&F control, PM-FOC torque control: HIL

The operational model considered here requires the following controller button settings:

- IM: `IM_spd_ctrl`: OFF
- IM: `IM_FOC`: OFF
- IM: `conv_on`: ON
- PM: `PM_spd_ctrl`: OFF
- PM: `PM_FOC`: ON
- PM: `conv_on`: ON

With these button settings, open-loop current & frequency control for the induction machine is enabled together with sensored field-oriented torque control of the PMsyn machine. This implies that the following set of users sliders are active:

- `IM_speed(rpm)`: Sets the synchronous shaft speed of the IM drive.
- `IM_id_ref(A)`: Control of the IM direct axis current.
- `IM_iq_ref(A)`: Control of the IM quadrature axis current.
- `PM_id_ref(A)`: Control of the PM direct axis current. Set to zero in FOC mode (as is the case here).
- `PM_iq_ref(A)`: Control of the PM quadrature axis current, which controls the shaft torque when in FOC mode (as is the case here).

Note that in this example the induction machine is controlled by a rotating (speed set by the IM speed slider) current vector defined as $\vec{i} = i_d + ji_q$.

An example of HIL drive operation, given in figure 8.50 is based on a IM speed slider setting of 1000 rpm, IM quadrature slider reference of 2 A and IM direct axis reference current 1 A. For the PM drive, the direct axis current reference is held at zero, while the quadrature current reference is set to -1.4 A.

Central to this figure is the Embed 'target execution module', which executes said out-file. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals. The plot given in figure 8.50 displays the per unit measured phase current

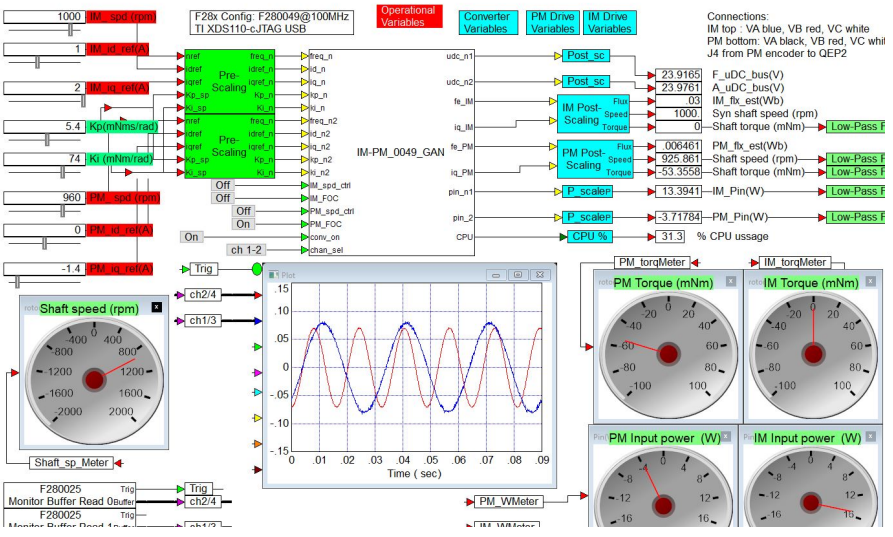


Fig. 8.50 HIL-Debug model, F0049-GANINV of a Dual IM-PM drive with open loop C&F control (IM) and FOC torque control (PM)

('blue') IM and phase current ('red') for the PM machine. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit LPF terminal vector voltage components u_α of both machines.

Observation of figure 8.50 confirms on the basis of the input power gauge readings that the IM machine is in motoring mode and the PM machine in generating mode. Furthermore, the PM shaft torque is negative and the IM shaft torque reading is set to zero (purposely in the controller) as FOC is not active for the IM. However, results to be discussed in the next subsection, show the presence of a friction torque component of ≈25 mNm. This implies that the steady-state shaft torque of the IM is ≈78 mNm for this operation mode (given the current settings used). Note also that the shaft speed is less then the synchronous speed (which confirms IM operation in motoring mode), hence the slip (calculated using expression (2.22)) is equal to 0.75 %. A

torque-slip characteristic of the induction machine can be found by changing the load (achieved by varying the quadrature current slider, positive and negative). Finally, note that the rotor flux estimate, shown on the numeric display (see figure 8.50) is NOT appropriate (in this mode), as said value is only applicable when IM FOC control is activate (to be discussed in the next section).

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

8.3.2.2 IM-FOC speed control, PM-FOC torque control: HIL

The operational model considered here requires the following controller button settings:

- IM: `IM_spd_ctrl`: ON
- IM: `IM_FOC`: ON
- IM: `conv_on`: ON
- PM: `PM_spd_ctrl`: OFF
- PM: `PM_FOC`: ON
- PM: `conv_on`: ON

With these button settings, field-oriented speed control for the induction machine is enabled together with sensored field-oriented torque control of the PMsyn machine. This implies that the following set of users sliders are active:

- `IM_speed(rpm)`: Sets the shaft speed of the IM drive.
- `IM_id_ref(A)`: Control of the IM direct axis current and (with FOC active) the rotor flux amplitude.
- `PM_id_ref(A)`: Control of the PM direct axis current. Typically set to zero in FOC mode (as is the case here).
- `PM_iq_ref(A)`: Control of the PM quadrature axis current, which controls the shaft torque when in FOC mode (as is the case here).

An example of HIL drive operation, given in figure 8.51 is based on a IM speed slider setting of 800 rpm and IM direct axis reference current 1 A, which corresponds to a rotor flux of 30 mWb. For the PM drive, the direct axis current ref is held at zero, while the quadrature current reference is set to -1.4 A. Central to this figure is the Embed 'target execution module', which executes said out-file. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals. The plot given in figure 8.51 displays the per unit measured phase current

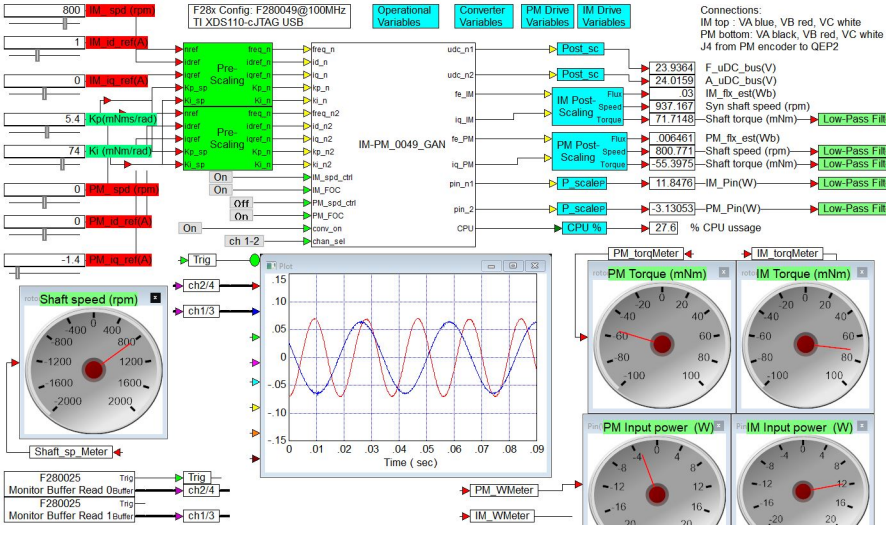


Fig. 8.51 HIL-Debug model, F0049-GANINV of a Dual IM-PM drive IM-FOC speed control and PM-FOC torque control (PM)

('blue') IM and phase current ('red') for the PM machine. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit LPF terminal vector voltage components u_α of both machines.

Observation of figure 8.51 confirms on the basis of the input power gauge readings that the IM machine is in motoring mode and the PM machine in generating mode. Furthermore the IM machine shaft torque is positive (as is the shaft speed) while the PM shaft torque is negative. Note that these two reading are NOT equal (in absolute value, as shown in the simulation model) due to the presence of a friction torque component of ≈25 mNm.
Note also that the synchronous speed is greater then the shaft speed (which confirms IM operation in motoring mode).

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

8.3.2.3 IM-FOC torque control, PM-FOC speed control: HIL

The operational model considered here requires the following controller button settings:

- IM: `IM_spd_ctrl`: OFF
- IM: `IM_FOC`: ON
- IM: `conv_on`: ON
- PM: `PM_spd_ctrl`: ON
- PM: `PM_FOC`: ON

- PM: `conv_on`: ON

With these button settings, field-oriented torque control for the induction machine is enabled together with sensored field-oriented speed control of the PMsyn machine. This implies that the following set of users sliders are active:

- `PM_speed(rpm)`: Sets the shaft speed of the PM drive.
- `IM_id_ref(A)`: Control of the IM direct axis current and (with FOC active) the rotor flux amplitude.
- `IM_iq_ref(A)`: Control of the IM quadrature axis current, which controls the IM shaft torque when in FOC mode (as is the case here).
- `PM_id_ref(A)`: Control of the PM direct axis current. Typically set to zero in FOC mode (as is the case here)

An example of HIL drive operation, given in figure 8.52 is based on a PM speed slider setting of 1000 rpm, IM direct axis reference current 1 A, which corresponds to a rotor flux of 30 mWb. The quadrature current reference for the IM is set to -0.6 A.
Central to this figure is the Embed 'target execution module', which executes said out-file. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals. The plot

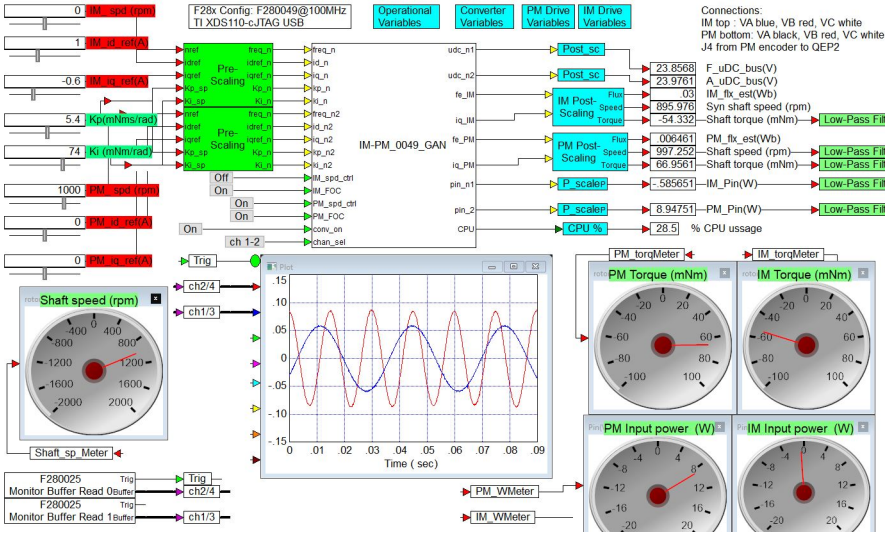


Fig. 8.52 HIL-Debug model, F0049-GANINV of a Dual IM-PM drive PM-FOC speed control and IM-FOC torque control

given in figure 8.52 displays the per unit measured phase current ('blue') IM and phase current ('red') for the PM machine. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit LPF terminal vector voltage components u_α of both machines.

Observation of the input power gauge readings (see figure 8.52) shows that the PM machine is in motoring mode and the IM machine is (marginally) negative. Furthermore, the PM machine shaft torque is positive (as is the shaft speed) while the IM shaft torque is negative. Note that these two reading are NOT equal (in absolute value, as shown in the simulation model) due to the presence of a friction torque component of ≈25 mNm. Note also that the synchronous speed is less then the shaft speed (which confirms IM operation in generating (plugging mode) mode).

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

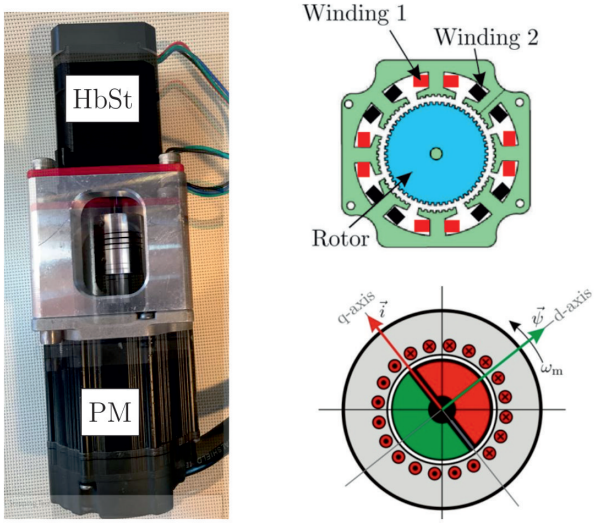
- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

8.4 Dual drive sensorless Hybrid Stepper sensored PM

The dual drive considered in this section consists of a mechanically connected PM synchronous machine [2] and bipolar hybrid stepper [9] as shown in figure 8.53. Included in this figure are two models of said machines, which shows that the stepper has two phases. The PM model has a vector diagram which shows the importance of controlling the current vector relative to the 'dq'

coordinate system in order to control the shaft torque. The same principles also apply to the stepper and both machines have a rotating flux vector, hence a rotating current vector is required (that is stationary relative to the flux vector) for each machine that is generated in the windings of each machine. An important difference between both machines is the pole pair number and the hybrid stepper-PM machine ratio is equal to 25:2 . This implies that the electrical rotational speed of the hybrid current/flux vectors are a factor 12.5 faster then the PM vector, for a given shaft speed.
For the PM machine, a shaft encoder is used to identify the flux vector, in terms of its orientation in the α, β stationary reference frame and align this with the controller 'd-axis'. For the Hybrid stepper a Texas Instruments sensorless algorithm FAST®is used, which makes use of the measured converter voltages/currents to identify the flux vector amplitude and orientation, as discussed earlier, see section 5.2.



(a) HbSt-PM Drive (b) Dual PM Hybrid stepper (HbSt) machines

Fig. 8.53 HbSte-PM dual drive setup

The overall drive concept of the HbSt-PM drive, as given in figure 8.54 shows the two controllers that generate the modulation indices for the respective converters that are connected to the machines in question. Note that the two phase hybrid stepper (with four leads, two for each phase) is connected to a three-phase converter as shown earlier, see figure 5.1.
An ADC unit is used to provide the currents, LPF converter voltages and encoder signal required for the two controllers. Control of the hybrid stepper is carried out with the aid of 'controller 1' and this makes use of the three

converter voltages and two phase currents. A module 'TransF.' is used to convert these variables to α, β coordinates for use by the FAST®algorithm, as discussed earlier, see section 5.2.1.1. Note that hardware used in this subsection is different to that used in said section, as will become apparent shortly. From a didactic perspective, the hybrid stepper is particularly interesting, as field oriented control is akin to a three-phase interior permanent magnet (IPM) 'flux-enhancement' machine, as discussed in chapter 9. This implies that typically, hybrid steppers can be viewed as salient PM synchronous machines (with $L_d > Lq$) and correspondingly, field-oriented control as discussed in section 9 is applied to the example stepper [9] discussed here.

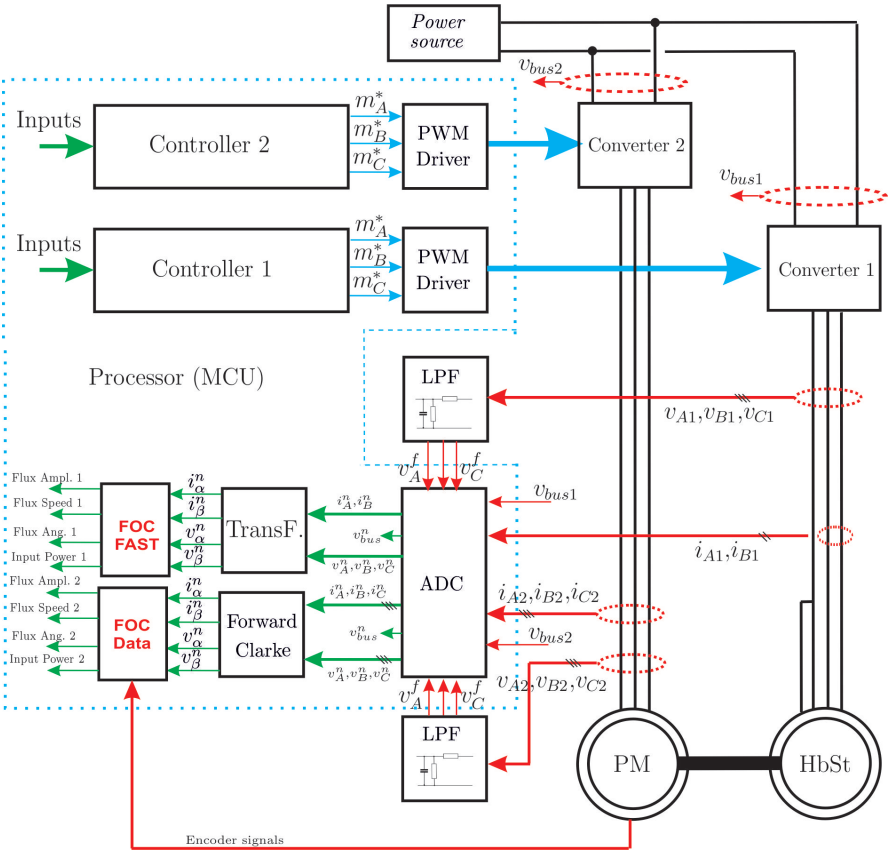


Fig. 8.54 HbSt-PM dual drive concept

Control for the PM synchronous machine is implemented in 'controller 2' and consists of: sensored field-oriented current control (see section 4.3.1).

8.4.1 Dual drive sensorless Hybrid Stepper sensored PM: PIL

Reference files:

- `Dual-PM-ST_FAST_F069-GANINV-PIL.vsm`
- `D-PM-HybSt-FOC_FAST-PIL_mod.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F28069M, which contains the DLL used to communicate with FAST®. This implies the use a target execution module for file: `D-PM-HybSt-FOC_FAST-PIL.out` as indicated in figure 8.55.

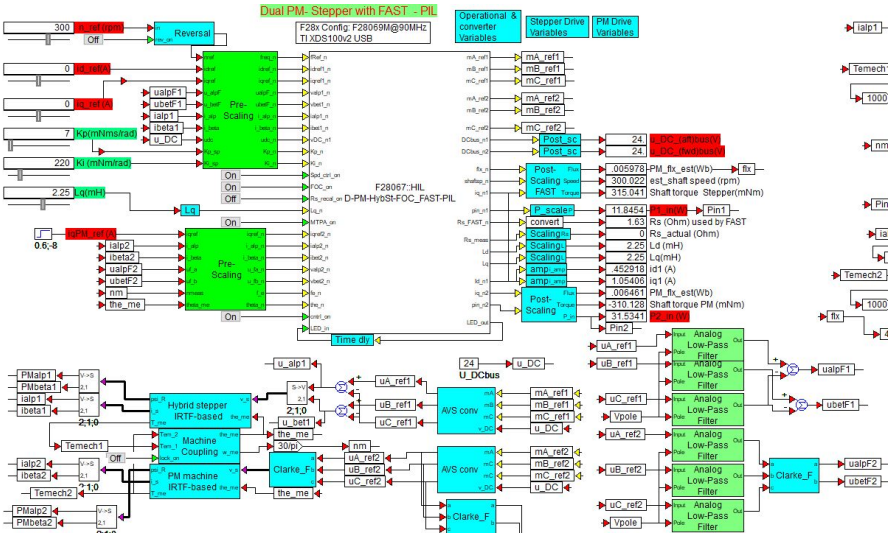


Fig. 8.55 Dual machine FOC sensorless operation of a hybrid stepper using FAST®with speed control connected to sensored PM machine with FOC torque control: PIL operation

Observation of figure 8.55 shows the presence of an IRTF based two-phase hybrid stepper module, which is connected to an IRTF based PMsyn machine via a `Machine Coupling` module. Said module represents a non-flexible coupling, that is defined by expression (8.2).

$$T_{\mathrm{me}}^{\mathrm{HbSt}} + T_{\mathrm{me}}^{\mathrm{PM}} = J_c \frac{d\omega_{\mathrm{me}}}{dt} \tag{8.2a}$$

$$\omega_{\mathrm{me}} = \frac{d\theta_{\mathrm{me}}}{dt} \tag{8.2b}$$

where J_c represents the combined inertia of both machines. The variables ω_{me}, θ_{me} represent the rotational shaft speed (rad/s) and shaft angle (rad) respectively. The shaft torque of the stepper and PM machine are represented by variables $T_{\text{me}}^{\text{HbSt}}$ and $T_{\text{me}}^{\text{PM}}$ respectively.

Two 'average voltage per sample' `AVS conv` modules acts as 'converters' for both machines. The six low-pass filters model the actual R-C filters present in both converters, as used to remove the PWM switching component. The LPF voltages `ualpF1`, `ubetF1` required for sensorless FOC stepper control are found using two `sum` units. Note that the direct and quadrature axis inductance values used by FAST®(see figure 8.55) are purposely set equal to the quadrature axis inductance value of this machine. The reason for this is that 'option 1', as discussed earlier (see subsection 5.2), is applied here for FOC sensorless control of a salient PM machine (which is a hybrid stepper in this case).

The target interface module that executes the out-file, generates the modulation indices required for the dual converter/motors and these components are not affected by use of the PIL. Generation of the out file is based on the PIL module (see figure 8.56), which must be compiled using the command: `toolbox>codegen`. Moving one level into the PIL module `D-PM-HybSt-FOC_FAST-PIL` shows a 'Drive Controller', see figure 8.57. Added to said controller is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that PIL control is operational.
Furthermore, the modules responsible for input power measurement `Pow_in`, with outputs `pin_n1` (for stepper) and `pin_n2` (for PM) are located outside the 'Drive Controller'.

Moving one level down into the `Drive Controller_` module, leads to a set of modules, given in figure 8.58. The `Current controller` deployed here, is different (when compared to three-phase examples discussed earlier), given the use of a two-phase machine, as was discussed earlier, see figure 5.22. The FAST®module `InstaSPIN FOC` provides per unit estimates of the rotorflux amplitude `ROTORFLX`, rotor flux angle `ang`, and electrical frequency `rotfreq`. Inputs to the FAST®module are the per unit measured LPF phase voltages and currents, as mentioned earlier. In addition, a variable `CurCTLstate` (generated in the `100 Hz` module) activates the FAST®module, which is also provided with the per unit measured DC bus voltage `Vdcn`. Also shown in figure 8.58 is a module `Rs_estimation`, which (when activated by the user with variable `Rs_ID`) estimates the stator resistance by injecting a direct axis current (amplitude set in the 'Operational variables' dialogue box). When stator estimation is active the angle output of the FAST®module is held to zero.

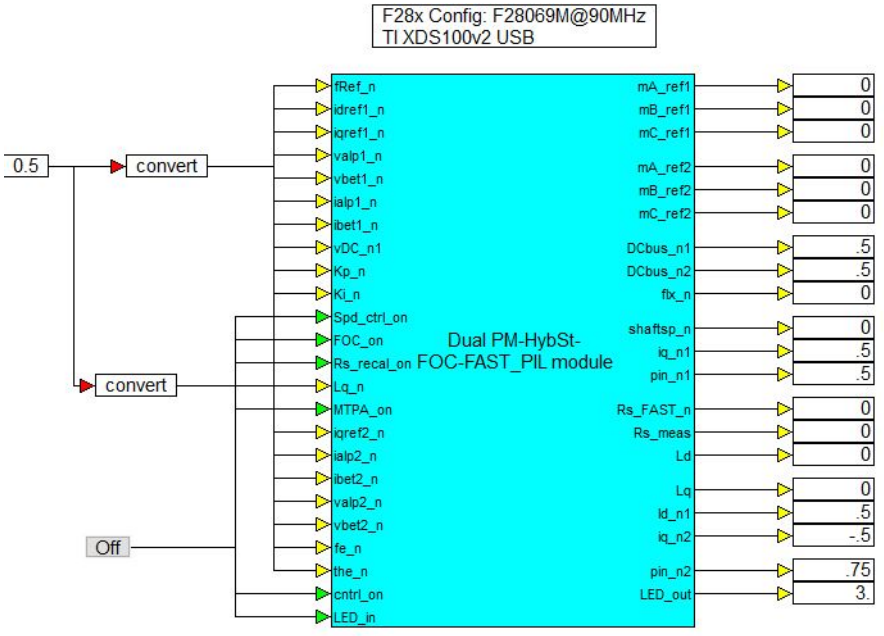


Fig. 8.56 Module:'D-PM-HybSt-FOC-FAST-PIL'

When FAST®is not used (encoderless off) the logic variable `encoderless_on` (see figure 8.58) is zero and this implies open loop current & frequency control is active. However, the FAST®module will also remain active under these conditions, hence an effective mode to test operation of this module.
The option 'Max Torque Per Amp' (MTPA), as is discussed in chapter 9, can be used to maximize the torque for a given current vector amplitude in IPM machines. A Hybrid stepper, can be viewed as a high pole pair IPM 'flux enhancement' machine, which implies that a MTPA module should be used to identify the optimum direct/quadrature current reference values for a given current vector amplitude generated by the speed controller. The module `IPM-Is Controller` shown in figure 8.58, generates the per unit reference direct/quadrature axis reference currents when the option `MTPA_on` is active `ON` (see figure 8.55). When this option is not used the speed controller simply controls the quadrature current reference (and sets the direct axis reference value to zero). The three-phase PM machine is used as a 'dynamometer', which implies the need for sensored FOC torque control as discussed earlier (see chapter 4.3). A module `PM FOC controller` as shown in figure 8.57 generates the modulation indices for the converter. Inputs to the module are the per unit measured currents `i_alp2`,`i_bet2`(converted from three phase using a Forward Clark transformation), encoder angle `the_e_n` and DC bus voltage `DCbus_n2`. Torque control is achieved via a slider or step function (as is the case for the PIl example, see figure 8.55) with sets the per unit quadrature

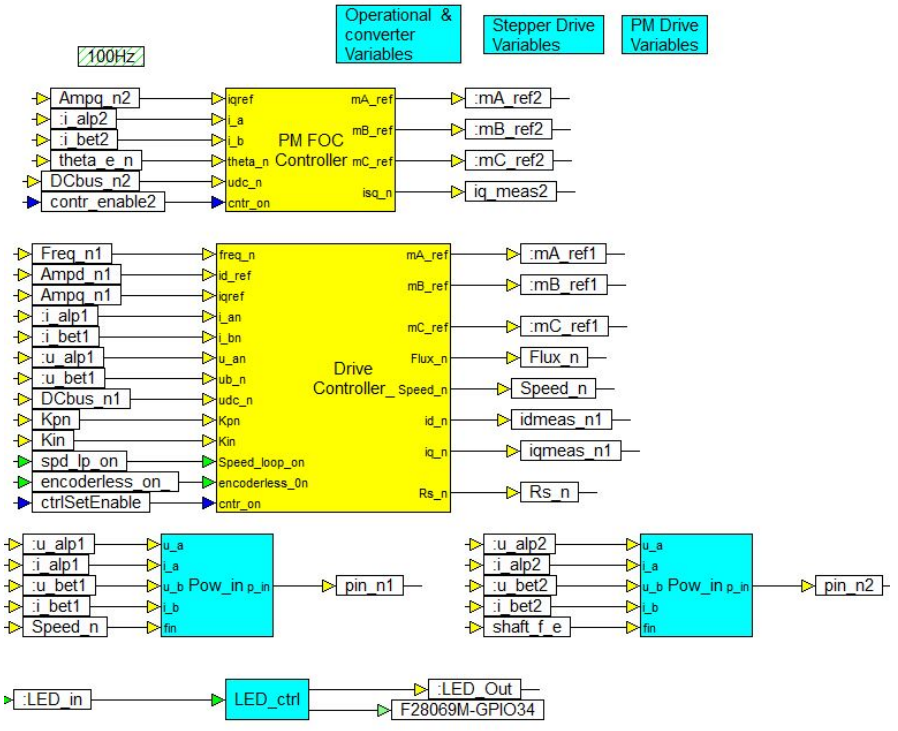


Fig. 8.57 Content PIL module, dual FOC torque/speed control of a hybrid stepper with FAST®connected to a FOC sensored PM machine

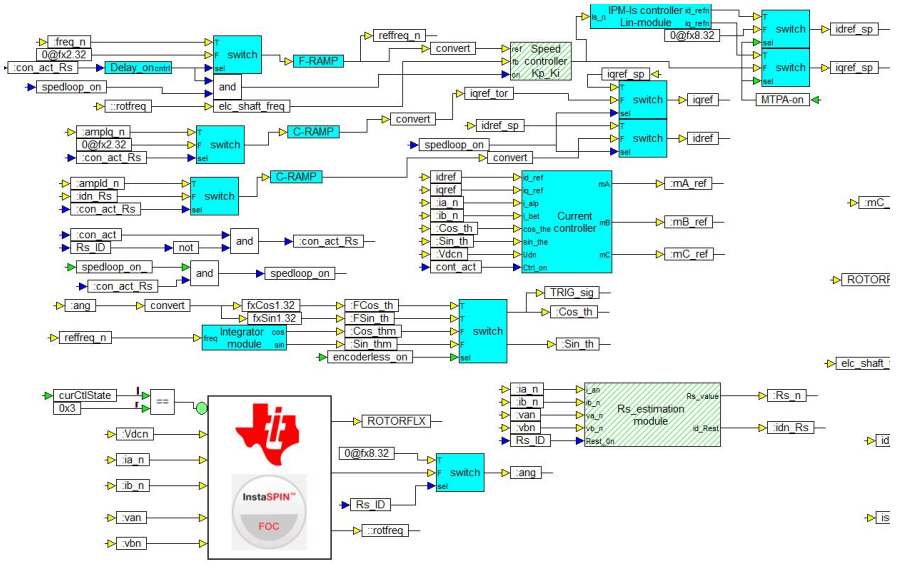


Fig. 8.58 Content 'PM FOC Controller' module

current reference variable `Ampq_n2`. The direct axis current reference for the PM machine is conveniently set to zero, as is apparent from the controller content, shown in figure 8.59.

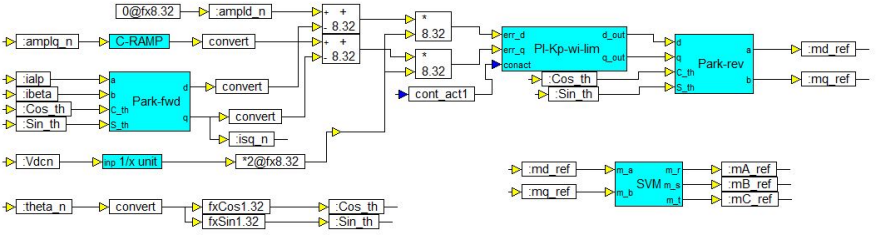


Fig. 8.59 Content 'PM FOC Controller' module

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational.

Operation of the stepper drive is set with a reference speed of 300 rpm and the dynamometer torque is initially set to zero. FAST® operation starts at 0.32 s and a stepper torque step is generated to allow the machine to accelerate to its steady-state value. Note that the scaled (by a factor 4.5k) FAST® estimated stepper flux settles to the PM flux value of the machine, as the direct axis current reference is held to zero (because the MTPA option is still OFF). At time mark 0.6 s a -8 A quadrature axis current for the dyno PM is applied, which corresponds to a load torque step, as may be observed from figure 8.60. The stepper, operating under speed control increases its shaft torque (because the speed controller generates a quadrature current component needed to maintain the reference speed). At time mark 0.80 s the MTPA function is activated, which implies that the direct and current reference currents are optimized to maximize the torque per amp ratio of the stepper. Evidence of this action is visible as the presence of a stepper direct axis component causes a increase in the estimated flux by 21 %. This implies that 21 % of the total stepper torque is now reluctance torque. For this machine (at the operating load point used in this example) the reduction in current amplitude (and input power stepper, after the MTPA is applied) is only marginal, but the operating principles of MTPA operation should be apparent by observation of the results. Also shown in figure 8.60 are the input power plots for both machines and perhaps surprising is the fact that the dyno input power remains positive, despite the fact that a negative output power situation is present when operating under load. This is because the dyno copper losses are substantial and exceed the negative output power level. The vector plots given in figure 8.61, which corresponds to time mark 1.0 s, are important as it shows operation of the stepper drive under MTPA control.

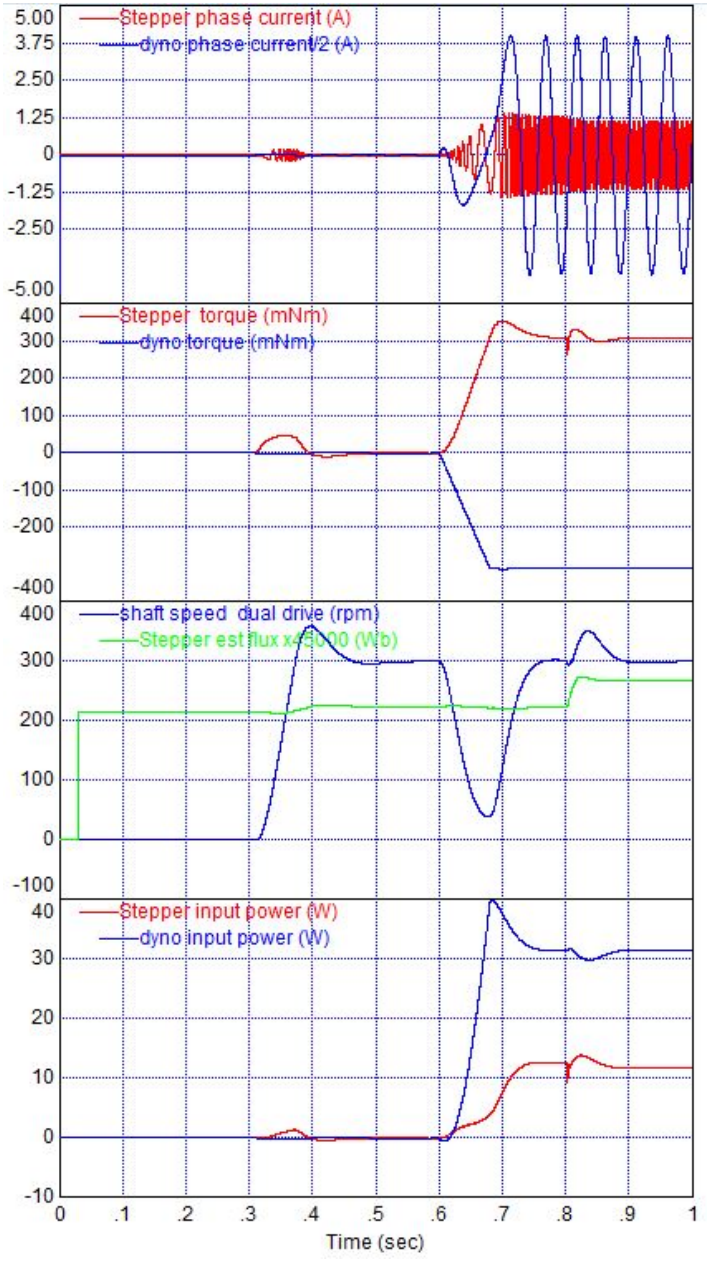
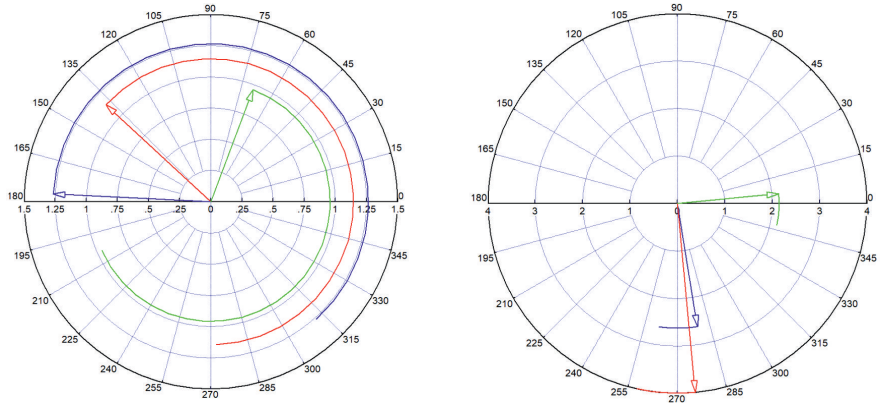


Fig. 8.60 PIL Result: dual machine FOC sensorless operation of a hybrid stepper using FAST®with speed control connected to sensored PM machine with FOC torque control

The vector plot for the stepper (figure 8.61(a)) shows the scaled (by a factor 200) rotor flux vector ('green'), current vector ('red') and scaled (by a factor 1/10) terminal voltage vector ('blue'). The presence of MTPA operation is also evident from the stepper vector plot, because the angle between flux and current vectors (in the d,q reference plane) is LESS then 90 deg. This is typical for IPM 'flux enhancement' machines, which is akin to operation of hybrid stepper drives, as these also exhibit saliency, with $L_d > Lq$.



(a) Stepper operation with MTPA active (b) PM dynamometer operation

Fig. 8.61 PIL result: dual machine FOC sensorless operation of a hybrid stepper using FAST®with speed control connected to sensored PM machine with FOC torque control: vector plots

The vector plot for the PM dynamometer (figure 8.61(b)) shows the scaled (by a factor 333) rotor flux vector ('green'), scaled (by a factor 1/2) current vector ('red') and terminal voltage vector ('blue'). The key observation from this PM dynamometer vector plot is that the rotor flux vector and current vector (which is due to the quadrature current) are orthogonal as required and expected. Note that the rotational speed of the stepper vectors are a factor 12.5 higher then those of the 'dyno' vectors, as may be observed when running the PIL example. This difference in rotational speed is due to the pole pair ratio of both machines, as was mentioned earlier.

8.4.2 Dual drive sensorless Hybrid Stepper sensored PM: HIL

Reference files:

- `Dual-PM-Stepper-F069-GANinv-compile.vsm`, for HIL-compile

- `Dual_PM_Stepper-F069-GANInv-debug.vsm`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL) shown in figure 8.62 consists of a `LAUNCHXL-F28069` module with two `GANINV` converters, which are electrically connected to two machines. The smaller (of the two) machines shown in figure 8.62 is a fifty pole pair two-phase hybrid stepper motor[9]. Said motor is mechanically connected to a four pole pair three-phase PM synchronous machine [2] with encoder, which is used as a dynamometer.

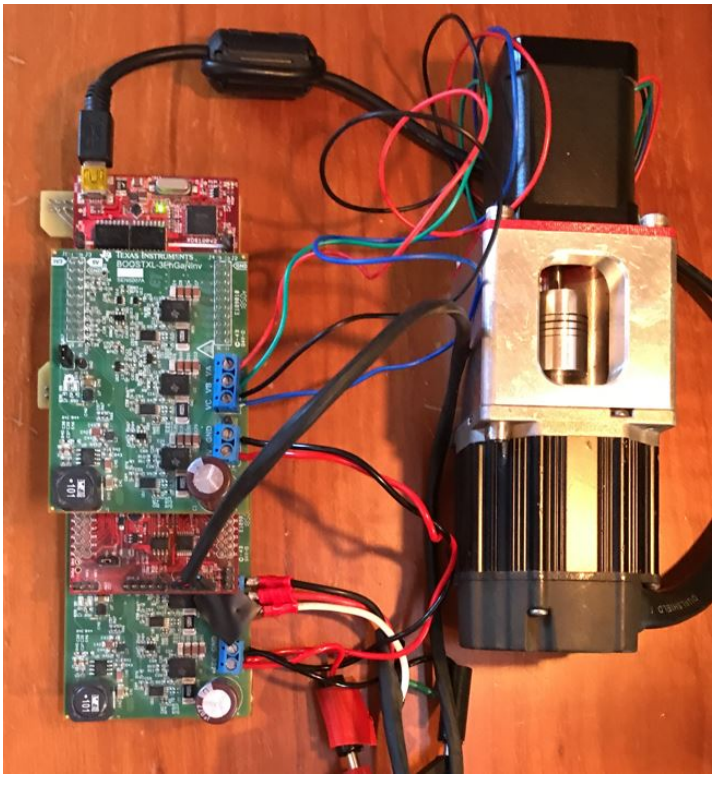


Fig. 8.62 Dual Hybrid stepper-PM drive for HIL operation

For HIL operation the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches. A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 8.63 shows the `FOC FAST Stepper-FOC sensored PM`

module, which must be compiled to generate an out-file. Inputs, shown in

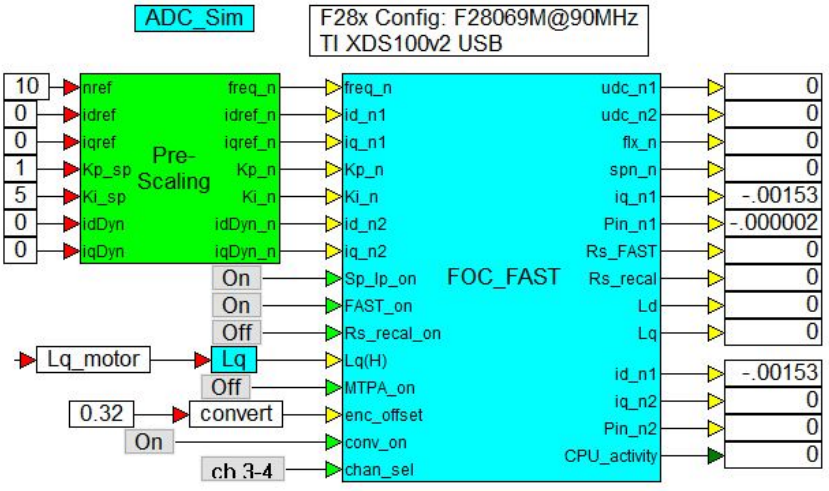


Fig. 8.63 HIL-Compile model of a dual hybrid stepper-PM drive

this stage as constants (for readability purposes) for this controller module are the stepper direct/quadrature axis reference currents `idref`, `iqref` and the reference shaft speed `nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller . Also added to this example is an input `Lq(H)` for tuning/setting the stepper quadrature axis inductance used by FAST®. Furthermore, two user inputs `idDyn`, `iqDyn` are provided that control the direct/quadrature axis currents of the sensored PM machine. Said quadrature axis current `iqDyn`, is used to control the torque of the PM synchronous machine. Also provided for the FOC sensored PM machine is an input `enc_offset`, which is used to align the 'd-axis' of said machine with the PM flux axis. Six logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed. The other four are used as follows:

- `Sp_loop_on`: Stepper speed loop control, when active the shaft speed slider controls the speed, based on the estimated shaft speed generated via FAST®.
- `FAST_on`: Activates sensorless FOC using FAST®for the stepper drive. Note when OFF the FAST®algorithm is still operational and this is useful when C&F control is active (both `Sp_loop_on` and `FAST_on` buttons: OFF).
- `Rs_recal_on`: Measures the stepper stator resistance at start-up (when `conv_on` is ON) by injection of a direct axis current (value set in the 'operational variables' dialogue box). The duration of the stator resistance

estimation time is set in the `100Hz` module via a time delay setting. It is important to verify that the estimation time is sufficient to reach a steady state resistance estimate.

- `MTPA_on`: Activates the 'Maximum Torque Per Amp' (MTPA) function for the stepper drive. When `ON`, the speed controller output is the reference current vector amplitude and this is used to generate the optimum direct/quadrature axis currents of the machine using expression (9.5). When `OFF` the speed controller outputs the reference quadrature axis current. Under these conditions the reference direct axis current is controlled by a user slider.
- `conv_on`: Activates the controller.
- `chan_sel`: Determines which two measurements are shown on the scope unit.

Outputs of the Controller module are:

- `udc_n1`, `udc_n2` : The per unit DC bus voltags of both converter modules, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- `flx_n`: The stepper 'total' flux estimate generated by FAST®, which in this case is the sum of PM flux and equivalent flux ψ_{eq}, as EMF estimation is based on the use of 'option 1', tied to expression (5.2).
- `sp_n`: A per unit shaft speed estimate `sp_n` generated via FAST®.
- `iqm_1`: The per unit stepper measured quadrature axis current, which makes use of the 'total' estimated flux and pole pair number to calculate the shaft torque.
- `pin_1`: A per unit estimate of the stepper input power.
- `Rs_FAST`: The per unit stepper phase resistance value used by FAST®.
- `Rs_recal`: The per unit stepper stator resistance value generated by the resistance estimator at start-up (when `Rs_recal_on` is ON). When selected the estimated stator resistance value is transferred to the FAST algorithm on completion of the estimation sequence. When not active the value shown remains zero and is not used by FAST®.
- `Ld`: The stepper direct axis inductance used by FAST®, which is equal to the sum of the quadrature inductance (set by the user slider) and incremental inductance ΔL (set in the dialog box). In this case, use is made op EMF estimation based on 'option 1' (see subsection 5.2), which implies that ΔL must be set to ZERO.
- `Lq`: The stepper quadrature inductance used by FAST®, controlled by the slider input.
- `idm_1`: The per unit stepper measured direct axis current.
- `iqm_2`: The per unit PM synchronous measured quadrature axis current.
- `pin_2`: A per unit estimate of the PM synchronous input power.
- `CPU_activity`: Provides the user with a % reading of MCU activity, relative to the controller sampling time.

Also shown in figure 5.26 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the Controller module ('right' mouse click, after selecting module) leads to the set of modules/dialogue boxes shown in figure 8.64. Three dialogue boxes are used, which contain the parameters

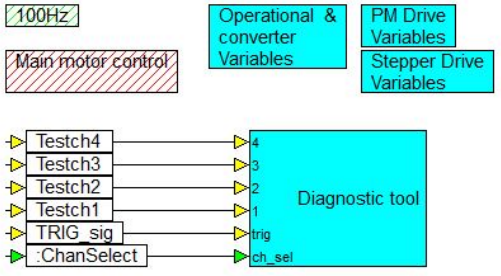


Fig. 8.64 One level down into the 'FOC FAST Stepper- FOC sensored PM' module

that need to be assigned by the user for HIL-operation. The 'operational & converter Variables' dialogue box, see figure 8.65, is used to assign full scale values for the voltage, current and frequency. In addition, the scaling as defined by the `ADC_fullscale_voltage/ADC_fullscale_current` used by the Analogue to Digital converter, as discussed earlier.
This dialog box also shows the low-pass filter corner frequency used by the converter for measuring the half-bridge voltages (see figure 4.12) and is an important parameter for FAST®, as it is used to compensate for speed dependant phase angle and amplitude errors of the LPF terminal voltage vector.
Also added to this dialogue box are a number of converter and FAST®related parameters namely:

- `PWM_sampling freq`: sets the ADC/controller sampling rate.
- `F_est/F_sampling`: set the sampling rate used by FAST®relative to the controller sampling rate, typically set to 1.
- `Speed filter pole`: corner frequency of the first order LP filter used by the stepper FAST®speed controller.

The 'Stepper Drives Variables' dialogue box, as given in figure 8.66, defines the current and frequency ramp rates used by the stepper drive. In addition, the PI current control gain and bandwidth for the stepper are defined in figure 8.66.

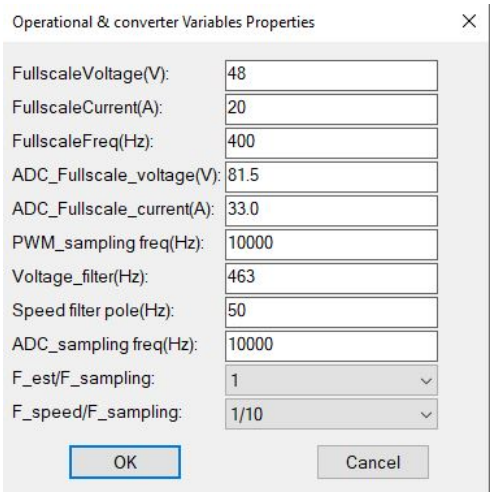


Fig. 8.65 HIL: Dialogue box 'Operational & converter Variables'

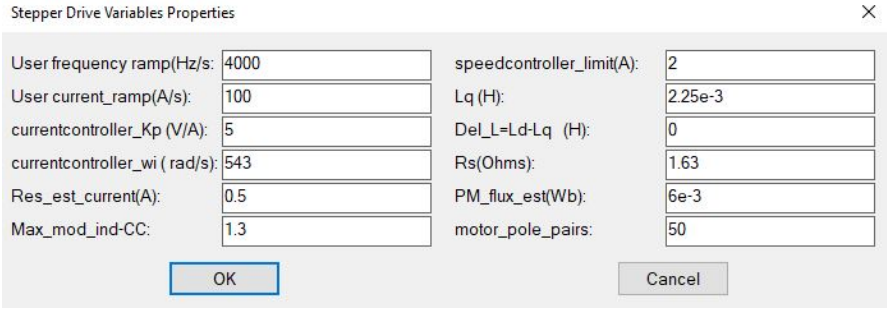


Fig. 8.66 HIL: Dialogue box 'Stepper Drive Variables'

Also added to this dialogue module are a number of motor/converter parameters namely:

- `Res_est_current`: sets the DC injection current used for estimating the stepper stator resistance, when the function `Rs_recal_on` is active.
- `Max_mod_ind CC`: sets the maximum modulation index for the current controller. When using a two-phase motor with a three phase converter (as is the case here) the theoretical limit is $\sqrt{2}$.
- `Speed controller _limit`: This is maximum current output value for the speed controller.
- `PM_flux)est (Wb)`: an important parameter which represents an estimate for the PM rotor flux in Wb. This value can be estimated, by using the drive in the C&F mode (see the 'Post Analysis' 4.2.1.5). For an IPM flux enhancement machine (like the hybrid stepper) the flux increases as a direct axis reference current is applied (with increasing load torque). Setting a

higher flux reference value (then the PM flux value) is beneficial for the FAST®dynamic flux range, i.e to allow tracking of the flux with increased load when MTPA is active.

- `Speed filter pole`: corner frequency of the first order LP filter used by the stepper FAST®speed controller.
- `Rs (Ohms)` the stator resistance value which is an important FAST®variable at start-up of the drive.
- `Lq`: the quadrature inductance, which can be estimated using an L-C-R meter (as required to set the current controller gain). The initial value used here, is replaced by the slider value when FAST®is operational.
- `Del_L=Ld-Lq`: the difference between the direct and quadrature inductance. This value must be set to zero , when using EMF 'option 1' estimation, despite the fact that a salient PM machine is used (see subsection 5.2).
- `p`: pole pair number, which can be found by operating the current in C&F mode and observing the rotational speed of the rotor when, for example setting the reference shaft speed to 60 rpm (one rotation per second).

The 'PM Drive Variables' dialogue box, as given in figure 8.67, defines the current and frequency ramp rates used by the PM synchronous drive. In addition, the PI current control gain and bandwidth for the dynamometer are defined in figure 8.67.

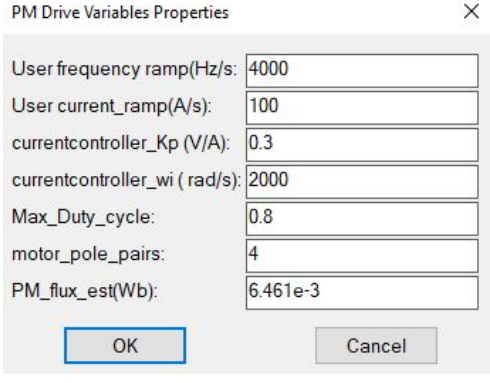


Fig. 8.67 HIL: Dialogue box PM Drive Variables'

Also added to this dialogue module are a number of motor/converter parameters namely:

- `Max_mod_ind CC`: sets the maximum modulation index for the current controller. When using a three phase converter with a three-phase machine (as is the case for the PM synchronous drive) the theoretical limit is $^2/\sqrt{3}$.
- `PM_flux)est (Wb)`: an important parameter, which represents an estimate for the PM rotor flux in Wb. This value can be estimated, by using the

drive in the C&F mode (see the 'Post Analysis' 4.2.1.5). The accuracy of this flux estimate is important as it is used to generate a dynamometer torque estimate value.
- `p`: pole pair number, which can be found by operating the current in C&F mode and observing the rotational speed of the rotor when, for example setting the reference shaft speed to 60 rpm (one rotation per second).

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a graph/plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables `Test1`, `Test2` or `Test3`, `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (State machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active. Furthermore, this unit generates a number of FAST®settings, which can generally be left unchanged.

The key component shown in figure 8.64 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit. Moving one level lower into the 'Main motor Control' module reveals a set of modules, shown in figure 8.68. Of these, the 'Drive controller' and 'PM FOC' controller has been discussed earlier, see figures 8.58, 8.59, where it was used in phase: PIL of the development tool-chain. Note that for the HIL example, an additional per unit direct axis reference variable `Ampd_n2` has been added to allow the user to control said current from a slider reference.
A module `Encoder speed/angle` generates the per unit electrical shaft frequency `shaft_f_e` and rotor angle `theta_e_n`, as required for sensored FOC control of the dynamometer. Inside said module a variable `enc_offset` is present, which is controller by a users slider. This function allows alignment of the controller 'd-axis' with the PM flux vector of the dynamometer.
Also shown in figure 8.68 are two power measurement modules `Pow_in` used for input power measurement of the stepper (output `pin_1`) and PM synchronous machine (output `pin_2`).
The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module. This module is configured with the following basic settings :

- PWM frequency: 10 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0 μs

- Board in use: `LAUNCHXL-F28069` with two `BOOSTXL-GANINV` converters

The converter dead-time is set to zero because gate drivers are present which handle this issue. This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed. The `ADC-PWM LaunchXL-069M-Dual 3phGANInv`

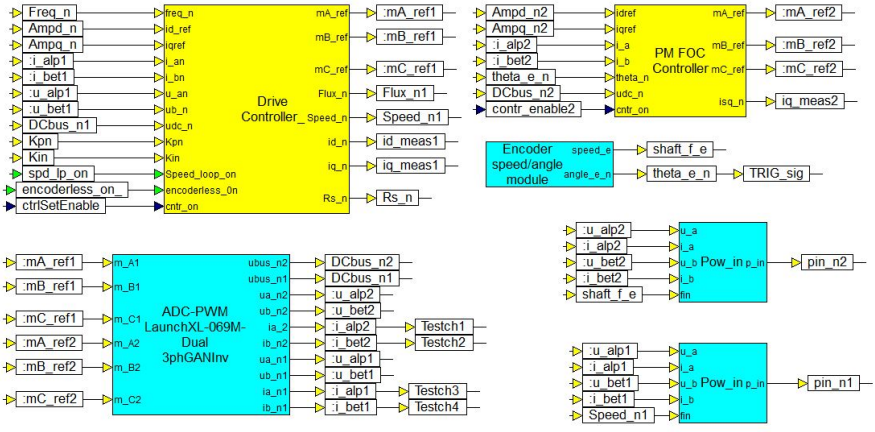


Fig. 8.68 One level down into the 'Main motor controller' module

unit is used generate the PWM signals for both converters. In addition, it is used to measure the six low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages. The measured phase currents of the two-phase stepper and three-phase PM synchronous machine are shown as per unit α, β outputs of the `ADC-PWM LaunchXL-069M-Dual 3phGANInv` unit.
ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the `100Hz` module) and controlled by said module, as mentioned previously. The per unit converter DC bus voltages: `DCbus_n1`, `DCbus_n2` completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example, shown in figure 8.69 and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. Three sliders are used to set the direct/quadrature axis current reference values and the electrical frequency of the stepper drive. A fourth slider is present to control the quadrature current of the PM synchronous machine. Proportional and Integral speed control gain settings for the stepper drive are also provided by two sliders. Post-scaling modules are used to convert

the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals. The scope given in figure 8.69 displays the per unit measured PM

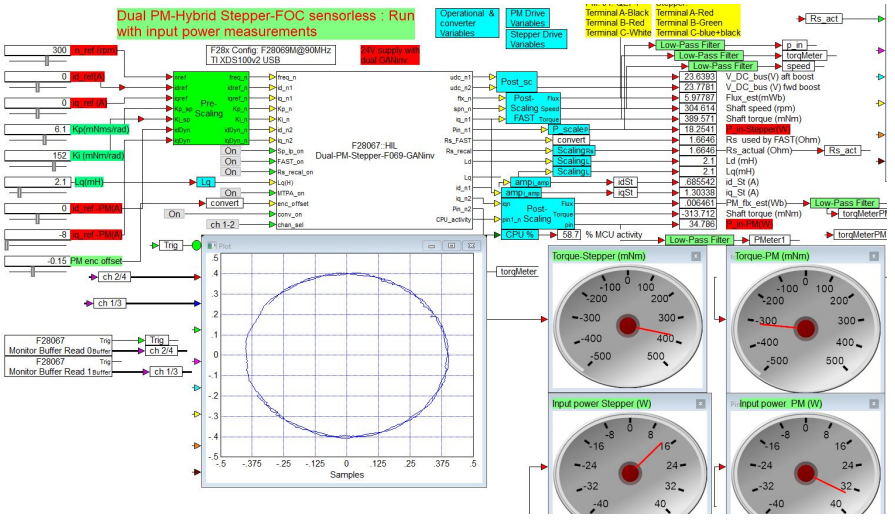


Fig. 8.69 Dual machine FOC sensorless operation of a hybrid stepper using FAST®with speed control connected to sensored PM machine with FOC torque control: HIL operation

synchronous current vector locus. This requires the channel select button to be set to `ch 1-2`. Selecting `ch 3-4`, will display the per unit current vector locus of the stepper. The actual amplitude of the current locus is found by using the 'Read Coordinates' function of the plot module, which shows a value of 0.4 , that must be multiplied by the 20 A full scale current which gives a value of 8.0 A. This corresponds to the quadrature axis slider setting for the PM synchronous machine, which confirms that the dual drive is operating under load conditions as used for the PIL example shown in figure 8.55. Operation of the stepper drive is with a reference shaft speed setting of 300 rpm, with MTPA and speed control active. Results shown on the numeric displays and gauges are in reasonable agreement with the PIL results, at time mark 1.0 s (see figure 8.60). A difference in absolute numerical display readings between PM and stepper is due a friction torque component in the experimental drive. A DC current probe with 'PicoScope' [20] was used to experimentally record the stepper phase current for drive operation as shown in the HIL example (figure 8.69). Observation of the PicoScope result (see figure 8.70) shows a current phase amplitude of 1.46 A and a waveform time period of 4.0 ms, which corresponds to a frequency f_{st} of 250 Hz. Correspondingly, the shaft speed of the stepper can be calculated using $n_{\mathrm{me}} = {}^{60 f_{\mathrm{st}}}/_{p_{\mathrm{st}}}$, where p_{st} is the

pole pair number of the stepper. This confirms that the stepper shaft speed is indeed 300 rpm.

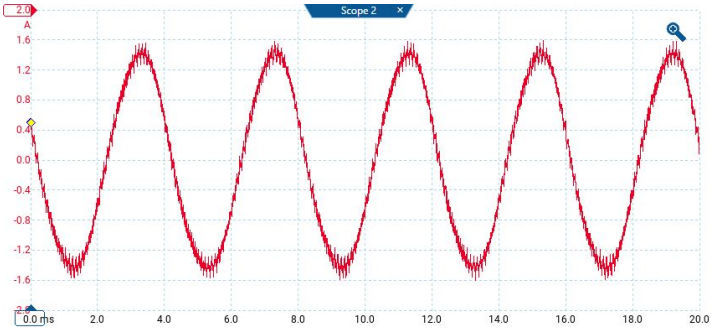


Fig. 8.70 Dual machine FOC sensorless operation of a hybrid stepper using FAST® with speed control connected to sensored PM machine with FOC torque control: measured stepper phase current

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 24 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

Post Analysis: FOC performance estimation
Reference file:

- `ST-IPM-FOC-limit-calculator.vsm`, for FOC stepper drive performance estimation.

The performance of a field-oriented drive, such as the hybrid stepper considered here is dependant on a number of critical parameters. For example the DC bus voltage and maximum modulation index used set a voltage amplitude locus, which can be used to calculate the maximum EMF that is available. This analysis is based on the use of figure 5.17 (shown for a IPM 'flux-weakening' example) and also requires knowledge of the direct/current axis currents and motor parameters. The reference file: `ST-IPM-FOC-limit-calculator.vsm` calculates the FOC speed limit, with inputs/outputs as shown in figure 8.71 and option 'Max Torque Per Amp' active for this stepper.
The HIL example shows a stepper current vector amplitude of 1.42 A (see figure 8.70) and use of this variable in the FOC performance estimator is shown in figure 8.71.

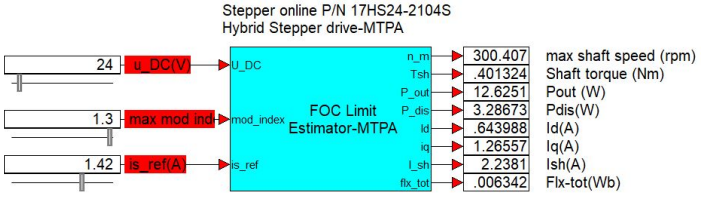


Fig. 8.71 FOC performance estimator of a hybrid stepper drive under MTPA control: operation with $i_s^{\text{ref}} = 1.42(A)$

These results show reasonable agreement with HIL and PIL in terms of the reference direct/quadrature axis currents, flux level and shaft torque. The FOC speed range for the input variables used, is in this case equal to 300.4 rpm. Hence operation of the HIL, with a reference speed of 300 rpm is at the limit of FOC operation. Furthermore, the PM synchronous machine was also at its rated value, which implies that FOC stepper operation at rated torque could not be shown with this experimental set-up.
However, the 'FOC limit estimator-MTPA' can be used to estimate operation under rated conditions, by setting the stepper current to 2.1 A (the rated stepper current), as shown in figure 8.72.
Observation of the numerical displays learns that the stepper FOC speed range is now reduced to 205 rpm. This is the result of increased direct/quadrature axis current levels. Notably, a larger direct axis current value leads to an increase in the total flux value of 7.5 mWb (note that the PM flux value is 4.7 mWb), which means that the machine is now operating with a higher EMF

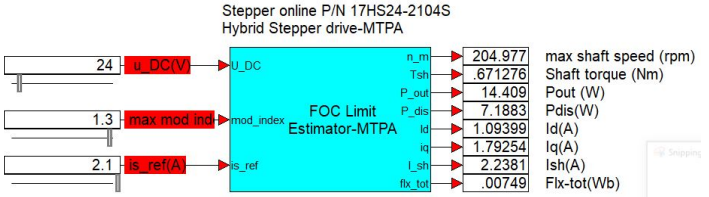
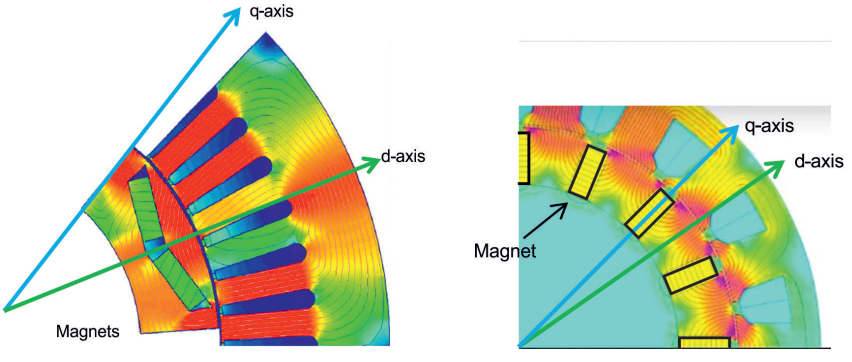


Fig. 8.72 FOC performance estimator of a hybrid stepper drive under MTPA control: operation with $i_s^{\text{ref}} = 2.1(A)$ (rated current)

and a reluctance torque component of 59 %, as can be calculated with the aid of expression (9.1). These observations again confirm that hybrid steppers are very similar to IPM 'flux enhancement' machines in terms of FOC control, as will become apparent in chapter 9.

Chapter 9
Interior Permanent Magnet (IPM) drives

'Interior Permanent Magnet' machines known as (IPM) are characterized by a rotor, which features 'buried' rotor magnets as mentioned earlier, see figure 2.15(a). They are closely tied to the PMsyn machine in terms of functionality and both use the same stator winding concept. However moving magnets inside the rotor changes the inductance-rotor angle profile. For PMsyn (also referred to as 'non-salient') the inductance is NOT a function of the rotor orientation. For IPM's, stator inductance is a function of the rotor angle with two extreme cases defined by the (d, q) axis inductance values, as shown in figure 9.1 for two types of IPM topology. The term $(L_d - L_q)$ known as



(a) IPM flux-weakening (b) IPM flux-enhancement

Fig. 9.1 IPM flux-weakening (a) [3] and flux-enhancement(b) [4] topologies

the 'incremental inductance' ΔL, is less than zero (for flux-weakening IPM), which is why a negative i_d is required to generate an equivalent flux ψ_{eq}, shown in equation (9.1) (given again here, for the sake of readability) below that supports the PM flux ψ.

$$T_\mathrm{e} = \frac{3}{2}\left(\underbrace{i_q\,\psi}_{T_\mathrm{ed}} + \underbrace{i_q\,\psi_\mathrm{eq}}_{T_\mathrm{eq}}\right) \tag{9.1a}$$

$$\psi_\mathrm{eq} = i_d\,\Delta L \tag{9.1b}$$

This generates an additional 'reluctance' torque component, which boosts the torque/volume ratio of IPM machines and improves efficiency, as will become apparent from an ensuing discussion based on the use of figures 9.2 and 9.3.

The perceived advantage of the flux-enhancement IPM is that the issue of 'flux-weakening' is readily resolved, as it simply requires a reduction of the (positive) direct axis current when the base speed is reached. Hence at high speed the reluctance torque component is then removed together with the EMF component $e_\mathrm{eq} = \omega_s\,\psi_\mathrm{eq}$ (which lowers the terminal voltage amplitude, thus allowing operation above the base speed)). Consequently, high speed operation is more efficient when compared to the (traditional) flux-weakening IPM, which requires an increase in the (negative) direct axis current when the base speed is reached.

The hypothetical question raised here is to identify the current vector amplitude and orientation ξ (see figure 9.2) for a given torque, arbitrarily set to 400 Nm in a flux weakening linear (ignoring magnetic saturation) IPM. Subsequent analysis using an optimization algorithm, to be discussed shortly,

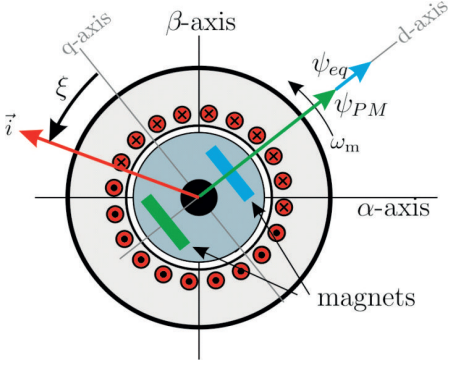


Fig. 9.2 Orientation ξ of the current vector to optimize the shaft torque in a flux-weakening IPM

are presented in figure 9.3(b). For comparison purposes the case $i_d = 0$ is

added and this shows that the current vector amplitude can be reduced from 218 A to 145 A, whilst maintaining the same torque. Consequently, the copper losses of the machine are reduced by a factor 55 %, which underlines the impact of adding a reluctance torque component to the IPM. This example is based on the IPM machine parameters of a Prius electric vehicle [3].

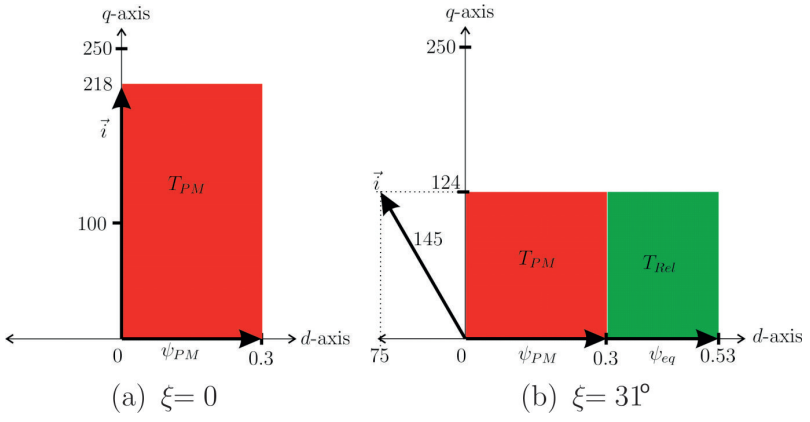


Fig. 9.3 Impact of adding a negative i_d component, case(a): $i_d = 0$, case(b) optimum i_d, i_q to maintain same torque.

IPM's are widely used for traction applications such as electrical vehicles. Furthermore, the presence of a stator inductance that is a function of rotor angle is helpful for overcoming the machine standstill issue for sensorless PM drives. However, control of IPM's is generally more complex because both direct and quadrature current values must be adjusted for a given reference torque or current amplitude (pending if speed or torque control is used, as will be discussed later) value. Furthermore, magnetic saturation complicates FOC control implementation, as will be discussed at a later stage.

For the magnetically linear case the process of determining the optimum ξ value that yields the highest shaft torque makes use of expression (9.1), together with a constraint that the current vector locus is a circle with given value i_s as shown in expression (9.2), which also shows the direct/quadrature current values as function of the angle ξ.

$$i_d = -i_s \sin \xi \tag{9.2a}$$
$$i_q = i_s \cos \xi \tag{9.2b}$$
$$i_s^2 = i_d^2 + i_q^2 \tag{9.2c}$$

Note that the angle ξ is defined as positive (which implies that the current vector is in the second quadrant in the d, q coordinate frame), as shown in example 9.2.
Substitution of (9.2a),(9.2b) into (9.1) gives:

$$T_e = \frac{3}{2} i_s \cos\xi \left(\psi - i_s \Delta L \sin\xi\right) \tag{9.3}$$

Partial differentiation of the electrodynamic torque with respect to the angle ξ gives:

$$\left(\frac{3}{2i_s}\right) \frac{\partial T_e}{\partial \xi} = \sin^2\xi - \left(\frac{\psi}{2i_s \Delta L}\right) \sin\xi - \frac{1}{2} \tag{9.4}$$

The optimum angle ξ^{opt} found with the aid of (9.4) and using the condition $\frac{\partial T_e}{\partial \xi} = 0$, which leads to expression:

$$\sin\xi^{\text{opt}} = \left(\frac{\psi}{4\Delta L i_s}\right) - \text{sign}(\Delta L)\sqrt{\left(\frac{\psi}{4\Delta L i_s}\right)^2 + \frac{1}{2}} \tag{9.5}$$

The corresponding reference currents i_d, i_q for the current controller (for a given current amplitude i_s) can then be found using (9.2).
For field-weakening IPM machines the condition $\Delta L < 0$ hold and a positive ξ^{opt} (current vector in the second quadrant) will occur. For non-salient machines expression (9.5) for the case $\Delta L \to 0$ which corresponds to an angle $\xi^{\text{opt}} = 0$.
At a later stage in this GUIDE synchronous reluctance drives will be discussed and the optimum ξ^{opt} angle for achieving MTPA operation can also be found using expression (9.5) with the parameters: $\psi = 0$ (no PM magnets), $\Delta L > 0$, which yields an angle of $\xi^{\text{opt}} = -45°$(the negative sign is due to the fact that the current vector is in the first quadrant). Hence, MTPA operation for synchronous reluctance machines is realised when the condition $i_d = i_q$ is maintained. This condition also applies to induction machines operating under FOC conditions, as discussed earlier, see equation (6.15).

The 'IPM-Is controller Lin-module' shown in figure 9.4, makes use of expressions (9.5),(9.2a) and (9.2b) (hence no look-up table is required) and generates the optimum direct/quadrature axis current i_d^{o}, as function of the current vector amplitude with parameters L_d, L_q and PM flux ψ. The corresponding quadrature current is then found using expression (9.2c), with $i_d = i_d^{\text{o}}$ and is also provided by said estimator given in figure 9.4 together with the

corresponding dialogue box. Input to this fixed point module is the (per unit)

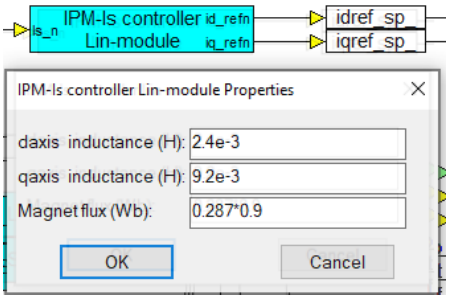
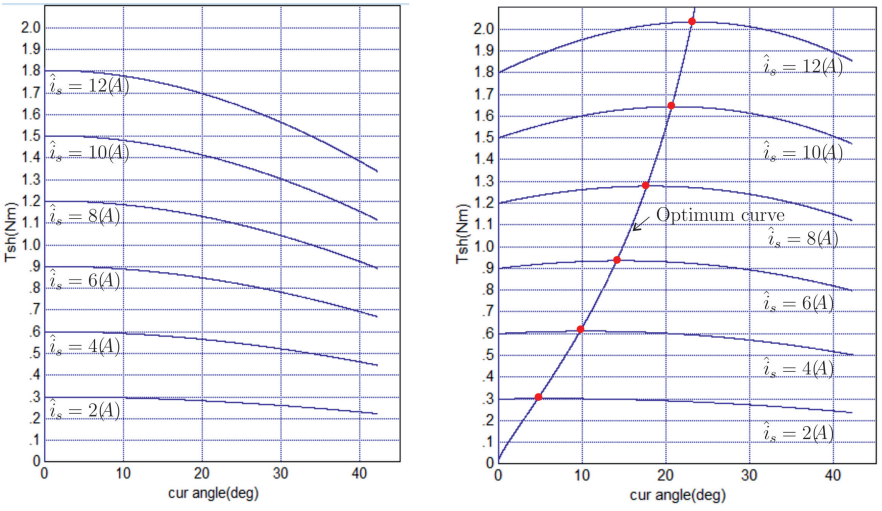


Fig. 9.4 'IPM-Is module' with dialogue box parameters of 'Test' machine

current amplitude `Is_n` and outputs are the (per unit) direct axis current & quadrature axis current reference variables `id_refn` and `iq_refn` respectively.

The process of verifying the effectiveness of this controller can be done by taking a 'test' machine and connecting this to a load (dyno) machine, as shown (for example) in section 8.2. Hence the machine operating under FOC torque control (in the dual control example) is replaced by the 'test' machine. The speed of operation is set by the attached machine to, for example, half rated speed e.g 500 rpm in this case and test machine FOC torque control is based on setting the current reference vector as shown in figure 9.2. The aim is to run the drive with a set of current vector amplitudes ranging from 2 A→12 A and (for each current value) varying the orientation of $\vec{i}$ over the range $\xi = 0 \rightarrow 45\,^{\circ}$. The result for the test motor considered here (detailed simulation to be discussed at a later stage, see section **??**) as given in figure 9.5 provides valuable insight. Added to figure 9.5 is the same machine WITHOUT saliency, hence $L_d = L_q$ and if a test sequence, as proposed here, is undertaken the result according to figure 9.5(a) is found. Consequently, for a non-salient machine choosing $i_d = 0$ is recommended to achieve 'maximum torque per amp' (MTPA) operation. When this test procedure is carried with a 'test motor' that does have saliency the result is as shown in figure 9.5(b). Added to this figure is the 'optimum curve' that is found using the 'IPM-Is module' (see figure 9.4), which also shows a dialogue box set with the parameters that correspond to those of the test machine). An important observation to be made from these results is that said machine is able to produce a given shaft torque of (for example) 1.8 Nm with a current vector amplitude that is reduced to 10.7 A (compared to the non-salient case, where 12 A is required). Therefore, the copper losses of the IPM test machine are 20% lower. A similar conclusion was also found for the IPM example discussed earlier (see figure 9.3). These results also confirm the validity and effectiveness of the 'IPM-Is module'.



(a) Non-salient PM (b) Salient PM

Fig. 9.5 Comparison between non-salient and salient (flux-weakening) drives

For IPM drive operation using FOC speed control the output of a speed controller is the current vector amplitude (which is set to be positive or negative (for speed reversal)). The speed controller has a current vector amplitude limit set by the user.
However, when using FOC torque control the shaft torque is set by the user, in which case the 'IPM-Is module' is placed in a control loop as shown in figure 9.6.

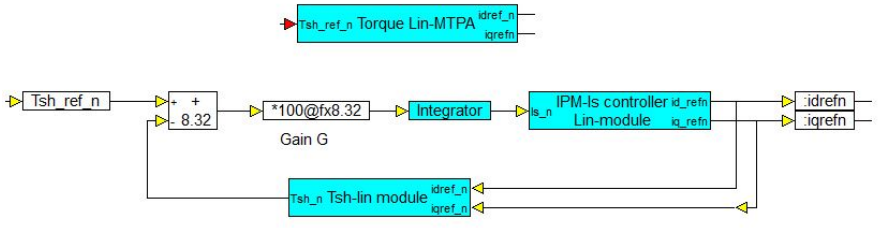


Fig. 9.6 'Torque Lin-MPSA module', linear case (no saturation) with MTPA capability

The module shown in figure 9.6 has as input the per unit shaft torque `Tsh_ref_n`, which is compared with the shaft torque calculated using a module `Tsh-lin`. Said unit is fixed point and is based on the use of expression (9.1), with $T_{\text{me}} = p\,T_e$. The integrator with gain `G` generates the required (per unit) current vector amplitude which corresponds to the condition: reference torque

equals measured torque. Output of this unit are the per unit optimized direct/quadrature axis currents.

Machines with rotors that utilize 'interior ' (buried) permanent magnets, as discussed earlier (see figure 9.1 and section 2.1.3.3) for traction applications, are typically prone to magnetic saturation effects and this is clearly observable in finite element based plots as shown in figure 9.7, where the 'red' areas reflect high flux density levels that significantly affect the flux-linkage/current characteristics of said machine. Consequently, torque control as discussed above for the linear IPM example, must be adapted to accommodate saturation effects, as will now be considered.

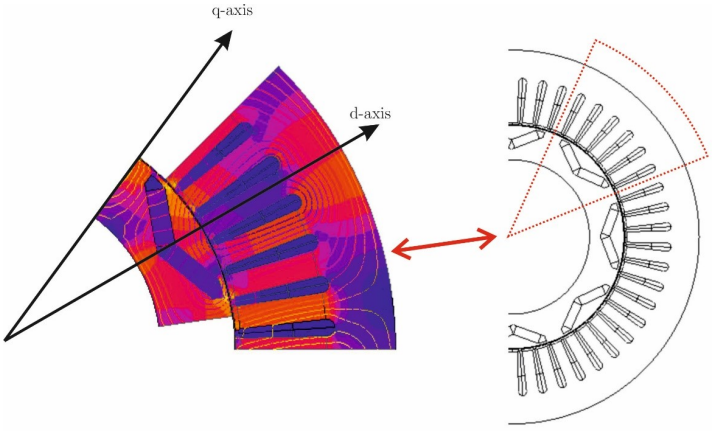
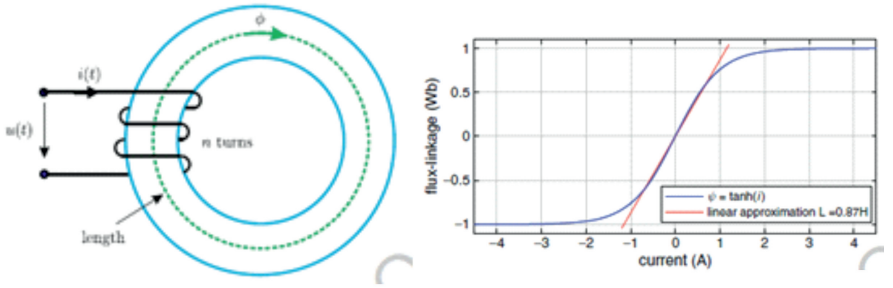


Fig. 9.7 Finite-element model of field-weakening 'FluxMotor' [19] IPM, used in the ensuing discussion

Prior to discussing the torque optimization strategy of an IPM with magnetic saturation it is prudent to briefly outline how these effects influence an electromagnetic unit like, for example, the toroid shown in figure 9.8(a). Central to this phenomenon is the non-linearity of said ferromagnetic materials, which is present in the form of a flux-density (B) versus magnetic field (H) characteristic known as a B-H curve. For flux density values which approach the saturated value B_{sat}, a linearity boundary is reached and magnetic field values increase significantly that leads to a non-linear flux-linkage/current characteristic $\psi(i)$ (see figure 9.8(b)). Within this curve a linear range is present, that defines the inductance of the toroidal example namely $L = \psi/i$. The behaviour of the toroid illustrates the impact that magnetic saturation has on the current. For example, assume the unit is connected to a sinusoidal voltage source, which drives the flux-linkage of the coil as depicted by the simulation model shown below (figure 9.9). In the linear case the inverse flux/current $i(\psi)$ relationship is simply $i = \psi/L$, which leads to the 'linear'



(a) Toroidal example (b) Flux-linkage-current characteristic

Fig. 9.8 Toroidal example and linear/non-linear flux-linkage/current characteristics [12]

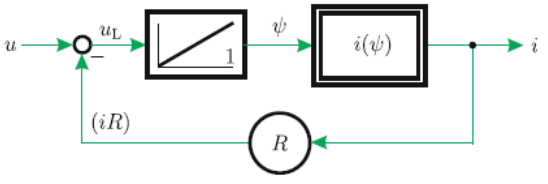


Fig. 9.9 Generic model of toroid with inverse flux-linkage/current characteristic [12]

result shown in figure 9.10. However, if an inverse model of the flux-linkage curve WITH saturation is used in a simulation model the results can be dramatically different, as may also be observed from figure 9.10 ('Saturation' current curve). The reason being that increased flux-linkage values lead to increased flux levels, which in turn causes the 'peaks' shown in the 'saturation' current plot (see figure 9.10). The toroidal example underlines the need for a

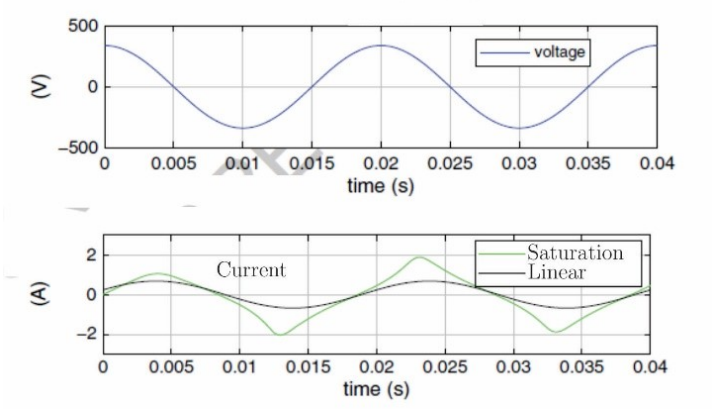
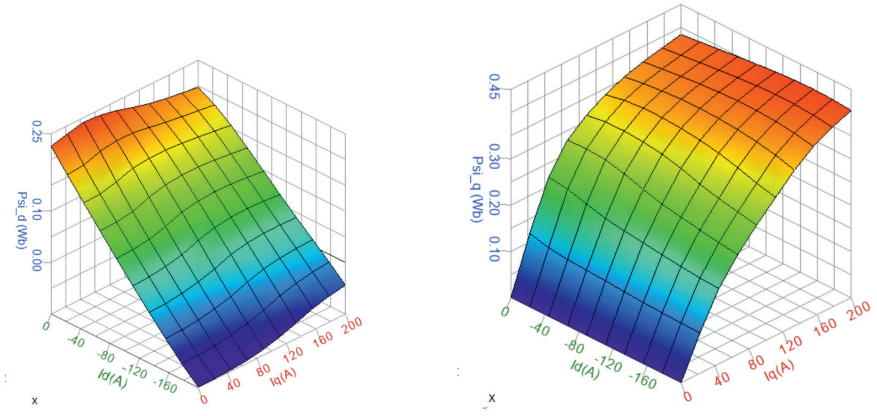


Fig. 9.10 Simulation results: Toroid model with/without saturation

simulation model that has access to the flux-linkage characteristic $\psi(i)$ and its inverse counterpart $i(\psi)$.

The issue of magnetic saturation and the need for inverse flux-linkage/current characteristics in an IPM IRTF model was mentioned earlier, see section 2.1.3.3. In this model a rotor based 'x,y' coordinate system was introduced, which is identical to 'd,q' axes used here. For the FluxMotor IPM example considered here, a set of flux-linkage/current characteristics was generated [19] (see figure 9.11), which clearly shows the presence of magnetic saturation. The



(a) Direct axis flux as function of (id, iq)

(b) Quadrature axis flux as function of (id,iq)

Fig. 9.11 Flux-linkage/current characteristics of the FluxMotor IPM

corresponding electrodynamic torque T_e of the IPM must now be calculated using the generalized torque expression [13], which is also used in IRTF models (see, for example, figure 2.18).

$$T_e = \frac{3}{2}\Im\left\{\vec{\psi}^*\,\vec{i}\right\} = \frac{3}{2}\left(\psi_d i_q - \psi_q i_d\right) \tag{9.6}$$

Added to expression (9.6) is the 'd,q' representation of flux-linkages and currents. For the linear case, the flux-linkages can be found using:

$$\psi_d = \psi + L_d i_d \tag{9.7a}$$
$$\psi_q = L_q i_q \tag{9.7b}$$

and substitution of expression (9.7) into (9.6) leads to the linear torque expression shown earlier (see (9.1)). The shaft torque $T_{\text{me}} = pT_e$ of the FluxMotor IPM for the magnetically linear and non-linear case can therefore be calculated using expression (9.1) and (9.6) respectively, where the latter makes used the flux-linkages/current plots shown in figure 9.11. The

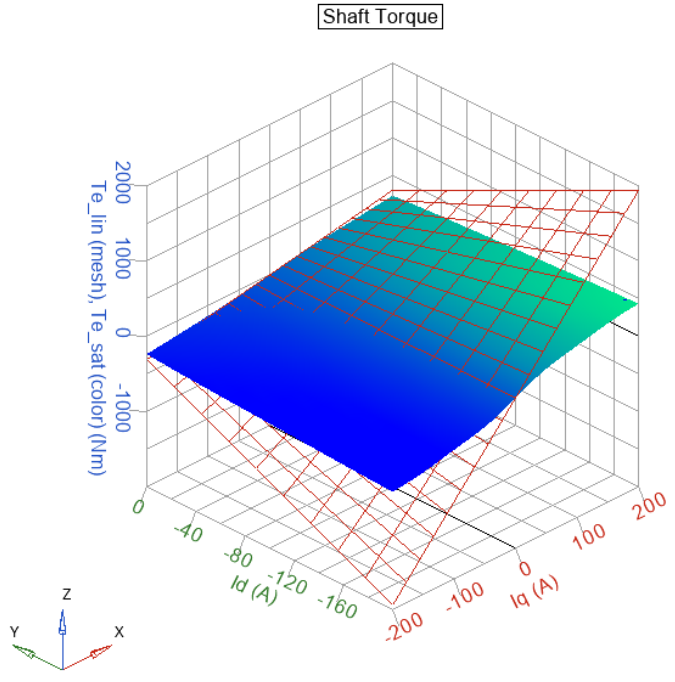


Fig. 9.12 Shaft Torque characteristics of the FluxMotor IPM with/without magnetic saturation

results, as given in figure 9.12 shows the shaft torque as function of i_d, i_q for the linear case ('red' grid) and non-linear case, i.e with magnetic saturation.

From a control perspective the aim is to find the 'Maximum torque Per Amp' (MTPA) for this IPM, which can be found by plotting the result $T_{\text{me}}/\sqrt{i_d^2+i_q^2}$ (based on the magnetic saturation result shown in figure 9.12), which leads to figure 9.13. Added to figure 9.13 is the optimum ('black' curve) `T/cur` function (which gives the highest shaft torque for the lowest current vector amplitude).

Observation of these results show the importance (for those cases where

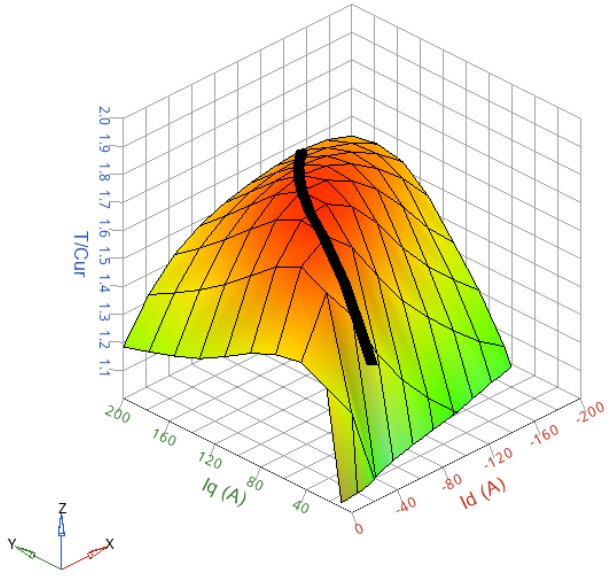


Fig. 9.13 Optimized torque/current (Nm/A) plot as function of i_d, i_q for the Flux-Motor IPM

linear models are inadequate) of using an IPM IRTF model with saturation (see figure 2.18), which raises the issue of finding the inverse current/flux-linkage characteristics of the IPM. A method to generate these characteristics is shown in figure 9.14 and this makes use of the flux-linkage/current plots shown in figure 9.11. Subsequent application of the inversion method shown

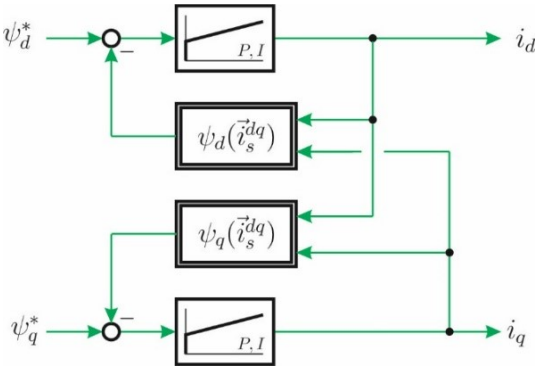
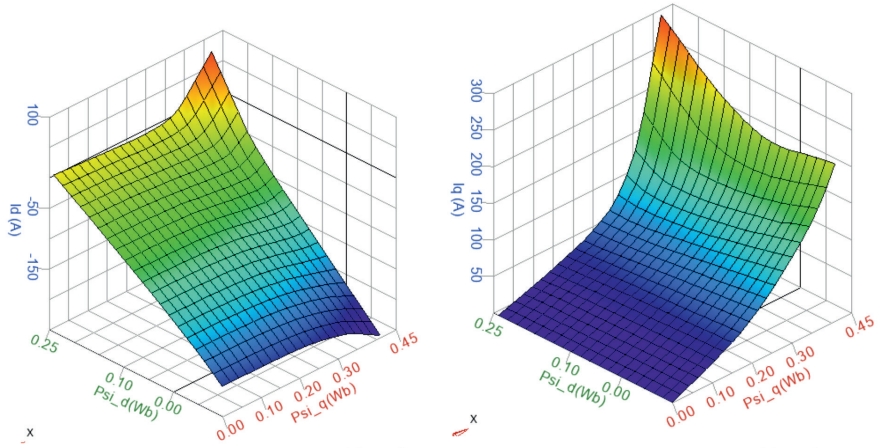


Fig. 9.14 Method that can be used for inverting the flux-linkage/current characteristics

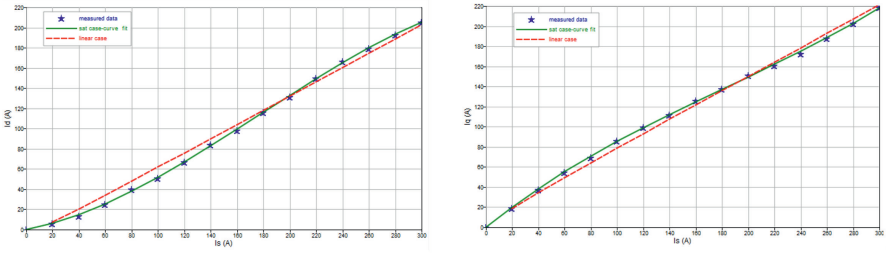
in figure 9.14 or using a more numerical approach [29] leads to the results shown in figure 9.15.



(a) Direct axis current as function of (ψ_d, ψ_q) (b) Quadrature axis current as function of (ψ_d, ψ_q)

Fig. 9.15 Inverse flux-linkage/current characteristics of the FluxMotor IPM

For IPM drives using speed control, a module is required which generates the optimum direct/quadrature axis current with the current vector amplitude as input. Such a model was introduced earlier (see figure 9.4) for the magnetically linear case. For IPM machine with saturation, such as the FluxMotor example considered here, the relationship $i_d(i_s), i_q(i_s)$ is derived either from finite-element results as shown in figure 9.13 or via measurement, using the approach discussed with the aid of figure 9.5. For the FluxMotor example the following results were found, see figure 9.16.



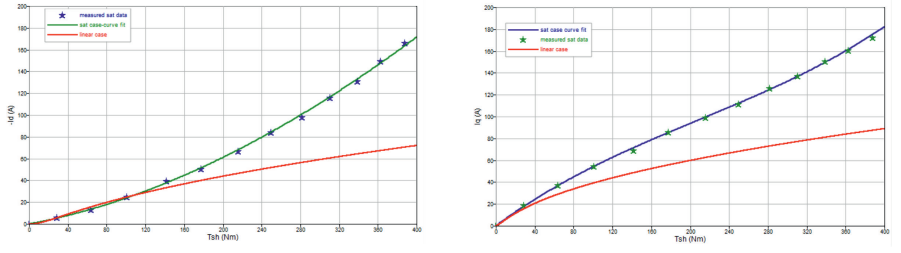
(a)i_d as function of current vector amplitude (b)i_q as function of current vector amplitude

Fig. 9.16 Direct/quadrature -axis current versus current vector amplitude of the FluxMotor IPM

Shown in this figure are the optimum direct/quadrature currents for the IPM

with/without ('red curve') saturation. The 'green' curves are third order polynomial functions which were fitted (using least-squares) to the measured IPM saturated IPM and are used (instead of lookup tables) in the Embed models to generate the direct/quadrature currents with i_s as input. Observation of these results shown that the curves with/without saturation are relatively closely aligned, which implies that optimum angle ξ (as introduced in figure 9.2) is not significantly different for the linear and saturated models. However, the current vector amplitudes are significantly different, as will become apparent from the ensuing allinea.

For IPM drives using torque control, a module is required with generates the optimum direct/quadrature axis current as function of the shaft torque. Such a model was introduced earlier (see figure 9.6) for the magnetically linear case. For IPM machine with saturation, such as the FluxMotor example considered here, the relationship $i_d\left(T_{\mathrm{me}}\right), i_q\left(T_{\mathrm{me}}\right)$ is derived either from finite-element results as shown in figure 9.13 or via measurement, using the approach discussed with the aid of figure 9.5. For the FluxMotor example the following results were found, see figure 9.17.



(a)i_d as function of shaft torque (b)i_q as function of shaft torque

Fig. 9.17 Direct/quadrature axis currents versus shaft torque of the FluxMotor IPM

Shown in this figure are the optimum direct/quadrature currents for the IPM with/without ('red curve') saturation. The 'green' curve in figure 9.17(a) and 'blue' curve in figure 9.17(b) are third order polynomial functions, which were fitted (using least-squares) to the measured IPM saturated IPM and are used (instead of lookup tables) in the Embed models to generate the direct/quadrature currents with shaft torque T_{me} as input. Observation of these results shown that the linear and non-linear curves are significantly different and these results were expected, based on the torque versus (i_d, i_q) plot shown earlier for the IPM FLuxMotor machine with/without saturation, see figure 9.12.

In the ensuing subsections the developmental tool-chain shown in figures 2.43, 2.45 is followed. For the IPM chapter the toolchain: 'Simulation' (SIM), 'Pro-

cessor in the Loop'(PIL),'CO-simulation' (COSIM) and 'Hardware in the Loop'(HIL) will be discussed for Texas Instruments (TI) based hardware examples. The drive control options considered are: Field Oriented Control' (FOC) sensored and sensorless (no encoder). Furthermore, Field weakening is also discussed for FOC sensored/sensorless examples. With respect to sensorless FOC, FAST[23] examples will be discussed.

9.1 IPM Sensored Field Oriented Control with encoder

The term 'field-oriented control' (FOC) implies that a synchronous 'd,q' coordinate system is tied to the PM flux vector of the IPM machine (see section 2.1.3.2). More specifically, the 'd-axis' of said coordinate system is coincidental with the flux vector $\vec{\psi}$, as may be observed from figure 9.18. To achieve the best possible torque for the lowest current vector amplitude (referred to as the Maximum Torque Per Ampere (MTPA) operating mode). This requires the use of a controller (as discussed in the IPM introduction) that identifies the appropriate orientation of the current vector in the 2nd quadrant (in this case, because assuming a 'flux-weakening' IPM topology) of the d, q coordinate system. The term 'sensored' FOC with encoder'(Enc),

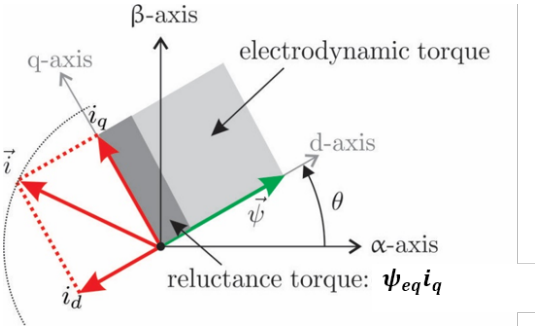


Fig. 9.18 Vector diagram: sensored field-oriented control (FOC)

implies that a shaft encoder is used to provide the angle θ needed to orientate the 'd-axis' relative to the magnetic flux vector of the IPM machine. For this type of application, an incremental shaft encoder is often used, which typically. In addition, the shaft encoder produces an index pulse (one pulse/per revolution), which provides the ability to use the incremental encoder for absolute angle sensing purposes (this implies that this function becomes active once an index pulse has been identified).

Note also, that in many field-oriented control examples open loop current & frequency control is often used (and added to the examples) as a method to check alignment of the controller 'd-axis' with the PM flux vector, when operating open loop (under no mechanical load conditions). Furthermore, this control is useful for identification of IPM machine parameters such as PM flux ψ and incremental inductance ΔL, as will be discussed.

The tool chain as introduced in figure 2.45 is partially used as a template for the ensuing subsections, here the toolchain: SIM→PIL→COSIM, will be considered, where use is made of the IRTF IPM FluxMotor [19] example introduced earlier, which can be configured with or without magnetic saturation.

9.1.1 IPM sensored FOC with Enc: SIM

Reference files:

- `IPM-FOC-Lin-Sat-SIM.vsm`
- `GA_inverse_dmap.csv`
- `GA_inverse_qmap.csv`

This example considers the simulation of a IPM FluxMotor [19] connected to an 'Average Voltage per Sample' converter, which is in turn connected to a controller, as shown in figure 9.19. The aim is to examine sensored FOC operation of the drive using a fixed point controller, that will subsequently be used in the PIL and COSIM examples.

Inputs to the controller are the direct/quadrature axis reference currents `id_ref`,`iq_ref`, shaft speed reference `n_ref` and shaft torque reference `T_ref(Nm)`. Three logic inputs are shown that provide the following functionality:

- `Ctrl`: converter on/off control.
- `FOC_on`: Field-oriented control active using encoder signal.
- `Spd_lp_on`: Field-oriented control active using encoder signal. Operation under 'speed control' when button ON and 'torque control' with button OFF.

When the logic inputs `FOC_on` and `Spd_lp_on` are both OFF, drive control is C&F and the direct/quadrature sliders control drive operation together with the shaft speed reference slider.

Note that when operating under torque control, the direct/quadrature axis current references are chosen to optimize torque production, i.e MTPA operation as discussed in the introduction. Next to the 'Drive controller' module

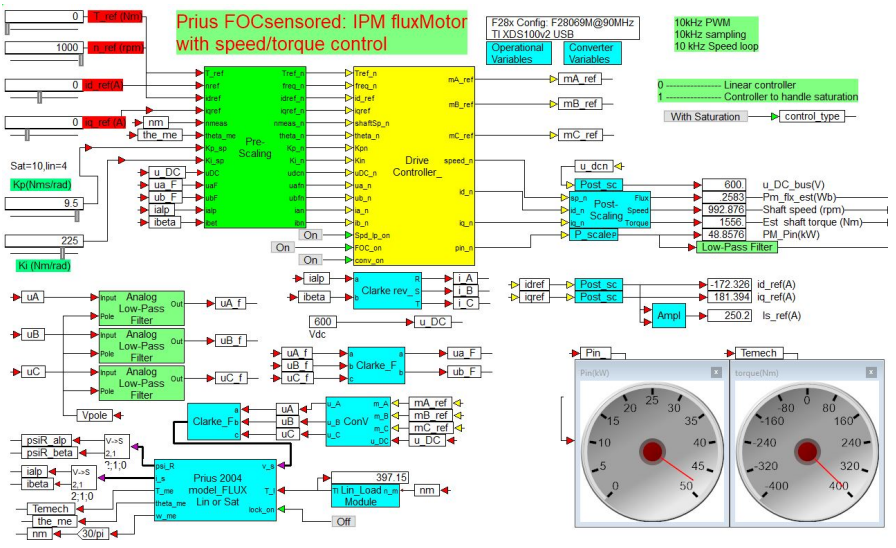


Fig. 9.19 Field-oriented torque/speed control IPM with encoder: simulation

an 'option' unit is given, which allows the user to select one of two options namely:

- option 0: `Linear`, which sets controller operation as required for a linear model of the FluxMotor IPM.
- option 1: `With Saturation`, which sets controller operation as required for a FluxMotor with magnetic saturation.

This basically means that the direct/current reference values as function of the shaft torque are set by the 'linear' plots shown in figure 9.17 (which were generated for the FluxMotor example used here) when option '0' is selected. Vice-versa when option '1' is selected, the plots 'sat-case curve fit', also shown in figure 9.17 are used by the controller.
When operating under speed control, the speed controller generates a vector current amplitude reference, which are converted in the controller to the optimum direct/quadrature reference values as shown in figure 9.16 (which are tailored to the FluxMotor example). Pending the choice of controller (option '0' or '1') the direct/quadrature current plots as function of current amplitude are selected appropriately. Hence when using a linear IPM model the 'linear' plots shown in figure 9.16 are used. Conversely, the 'sat case curve fit' plots should be used when a IPM with saturation is selected. The aim of the controller is to identify the required current vector amplitude to meet the criteria: measured shaft speed equals user defined reference speed.

The FluxMotor module `Prius 200f model FLUX lin or Sat` is an IRTF

based IPM representation. This module provides the user with two options namely:

- `Linear`: When selected an IRTF model as shown in figure 2.17 is used with the linearised FLuxMotor parameters
- `Saturation`: When selected an IRTF model as shown in figure 2.18 is used with the inverse flux-linkage current characteristics shown in figure 9.15, which are applicable the the FluxMotor example used here.

This implies that the user has the ability to use a MTPA controller and/or IPM module, which can accommodate saturation effects.
Also shown in figure 9.19 are three LPF `Analog Low-pass filter` modules that are used in the generic converter to remove the PWM component. Outputs of the forward Clarke converter variables `ualpF`, `ubetF` are used by the controller to calculate motor input power.
Outputs of the controller are three modulation indices, that control the three converter half bridges. The voltages supplied to the motor `u_A`, `u_B`, `u_C` (see figure 4.176) are found by multiplying the modulation indices `mA_ref`, `mB_ref`, `mC_ref` by half the DC bus voltage as executed in module `ConV`, as mentioned earlier. These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables. The aim of the switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sub chapter 2.1.5).
A linear load is introduced, which provides a value of 400 Nm at a shaft speed of 1000 rpm.

Moving inside the `Drive Controller` reveals the structure shown in figure 9.20. Readily apparent is the synchronous 'current controller' module, as

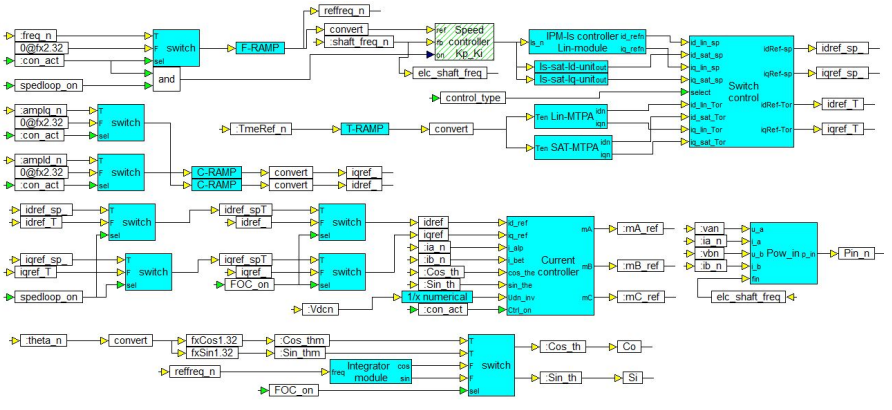


Fig. 9.20 IPM FOC speed/torque controller structure

discussed earlier, see figure 4.125. Both proportional gain, bandwidth and the

maximum modulation index of said controller must be provided in the 'operational variables' dialogue box. The speed controller module, as introduced earlier, see section 3.3, must be provided with appropriate proportional and integral gain values, that depend on the combined load inertia and user defined speed control bandwidth (see equation (2.4)). A set of sliders is introduced in figure 9.19 for this purpose.
Output of the speed control unit is the scaled current vector amplitude which is connected to modules `IPM-Is controller-Lin module` and the modules `Is-SAT-Id unit`,`Is-SAT-Iq-unit`. When the controller option '0' is selected, module `IPM-Is controller-Lin module` is active the scaled direct/quadrature currents of the `switch control unit`, are those generated said module. The content of module `IPM-Is controller-Lin module` is reflected by equations (9.5), (9.2) and the 'linear' curves shown in figure 9.16. When controller option '1' is selected, the `Switch control` unit takes the inputs from the two modules `Is-SAT-Id unit`,`Is-SAT-Iq-unit` and passed onto variables `idref_sp`, `iqref_sp`. The modules `Is-SAT-Id unit`,`Is-SAT-Iq-unit` represent third order polynomials, as shown by the curves 'sat-case curve fit' in figure 9.16.

When the option 'torque control' is selected the output variables `id_ref_T`, `iq_ref_T` (of the module `Switch control`) are used by the synchronous current controller. The scaled shaft torque, variable `Tme_ref_n` (value set by the `T_ref` slider in figure 9.19) is used by modules `Lin MTPA`,`SAT MTPA`. Module `Lin MTPA` (as discussed earlier, see figure 9.6), will be active when controller option '0' is selected, in which case the outputs are as shown by the 'linear' curves in figure 9.17. If there is a need to represent saturation, when using torque control, controller option '1' must be used, in which case the outputs of the module `SAT MTPA` are used by the synchronous current controller. The content of the module `SAT MTPA` are third order polynomials represented by the curves 'sat case curve fit' in figure 9.17.

9.1.1.1 Results, IPM sensored FOC with Enc: SIM-linear

This subsection considers the following options for the controller and IPM machine:

- Controller mode: `Linear Controller`
- IPM machine model: `Linear`

Hence operation where the controller is matched to the linear IPM machine model and operation results for torque and speed control are presented.
The results shown in figure 9.21, represents operation under torque control with T_{ref} =300 Nm, hence the steady state shaft speed is dependant on the applied load torque. Note that at start up ramp functions are used, which determine the rate of reference currents/shaft speed and shaft torque changes,

as set in the 'operational Variables' dialogue box. An example with speed

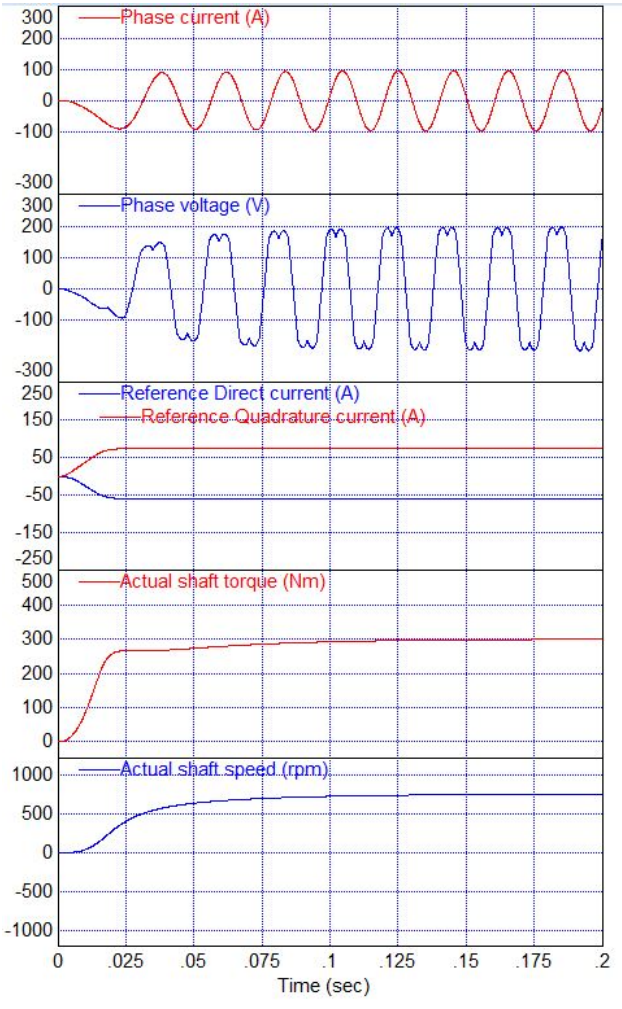


Fig. 9.21 SIM result, IPM operation, FOC-torque control active: mode SIM-linear

control active, as given in figure 9.22, shows that around that time mark 100 ms the reference speed (set to 1000 rpm) is reached. Correspondingly, the speed controller provides the direct/quadrature currents (consistent with MTPA operation) needed to generate a shaft torque that matches the load torque. A linear load torque/speed characteristic is assumed, as discussed earlier, which is set to produce 400 Nm at 1000 rpm. These results confirm that the required speed with matching shaft torque is provided by the drive. Of importance is to consider the vector plot, see figure 9.23 at time mark: 0.2 s,

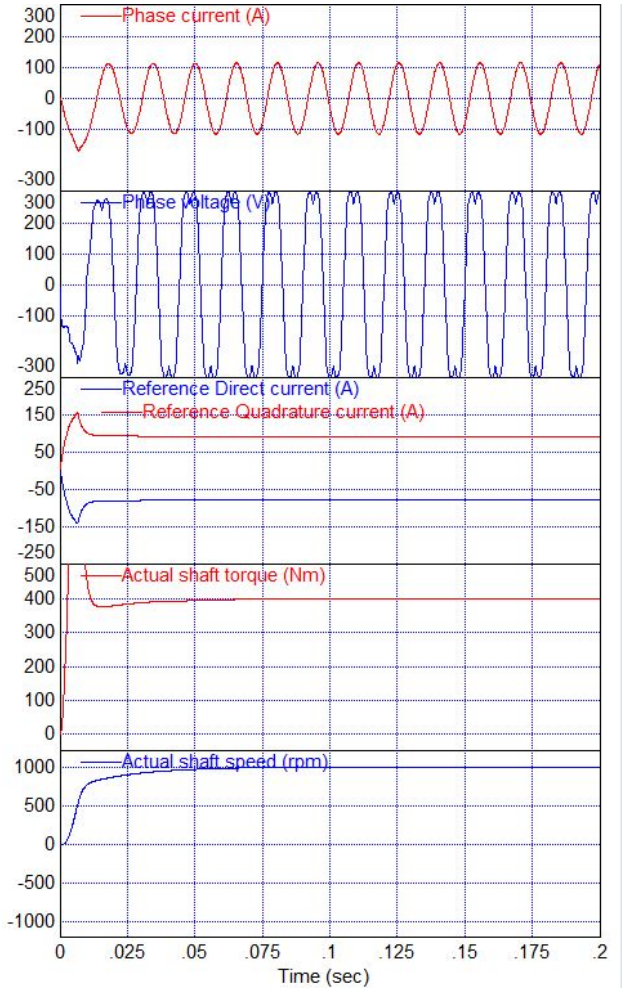


Fig. 9.22 SIM result, IPM operation, FOC-speed control active: mode SIM-linear

which shows the orientation of the scaled (factor 100) unity d-axis ('green'), voltage vector ('blue') and the current vector ('red'). Observation of the voltage vector amplitude learns that operation is close to the maximum limit $u_s^{\max} = {}^2/\sqrt{3}\,300 = 346$ V set by the current controller.

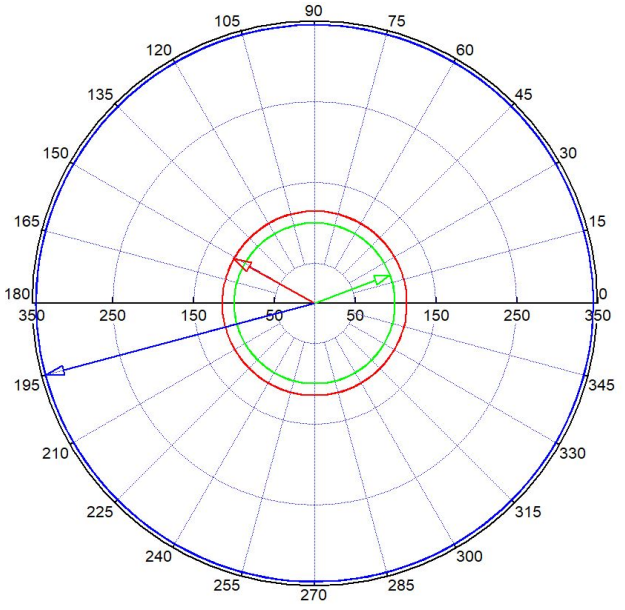


Fig. 9.23 SIM result, IPM operation, FOC-speed control active, vector plot: mode SIM-linear

9.1.1.2 Results, IPM sensored FOC with Enc: SIM-saturation

This subsection considers the following options for the controller and IPM machine:

- Controller mode: `With saturation`
- IPM machine model: `saturation`

Hence, operation where the controller is matched to the IPM machine model, with 'deep' magnetic saturation and operation results for torque and speed control are presented.
The results shown in figure 9.24, represents operation under torque control with T_{ref} =300 Nm, hence the steady state shaft speed is dependant on the applied load torque. Note that at start up ramp functions are used, which determine the rate of reference currents/shaft speed and shaft torque changes, as set in the 'operational Variables' dialogue box. An example with speed control active, as given in figure 9.25, shows that the speed controller current limit 300 A (set in the operation dialogue box) is initially reached and around that time mark 100 ms the reference speed (set to 1000 rpm) is reached. Correspondingly, the speed controller then provides the direct/quadrature currents (consistent with MTPA operation needed to generate a shaft torque that matches the load torque. A linear load torque/speed characteristic is assumed, as discussed earlier, which is set to produce 400 Nm at 1000 rpm. These results confirm that the required speed with matching shaft torque is provided by the drive. Of importance is to consider the vector plot, see figure 9.26 at time mark: 0.2 s, which shows the orientation of the scaled (factor 100) unity d-axis ('green'), voltage vector ('blue') and the current vector ('red'). Observation of the current vector amplitude learns that it is substantially increased (compared to the linear case, figure 9.23) due to 'deep' saturation of the IPM machine. The short-circuit current of this machine, defined as $i_{\text{sc}} = \psi/L_q$ =28 A is low compared to the 251 A current vector amplitude required for this operating point. Consequently, the voltage across the inductance component u_{Lq} has also substantially increased, which leads to a reduction in the terminal voltage amplitude, as is also apparent from figure 9.26. It is also noted that orientation of the current vector relative to the controller d-axis has not significantly changed, when compared to the linear case. This was expected given the results shown in figure 9.16. These show that curves for the 'linear' and 'saturated' MTPA case are in close proximity, hence the orientation of the current vector (relative to the controller d-axis (and PM flux)) will therefore, be approximately the same. However, the operating point (defined by the current vector amplitude) will be considerably different.

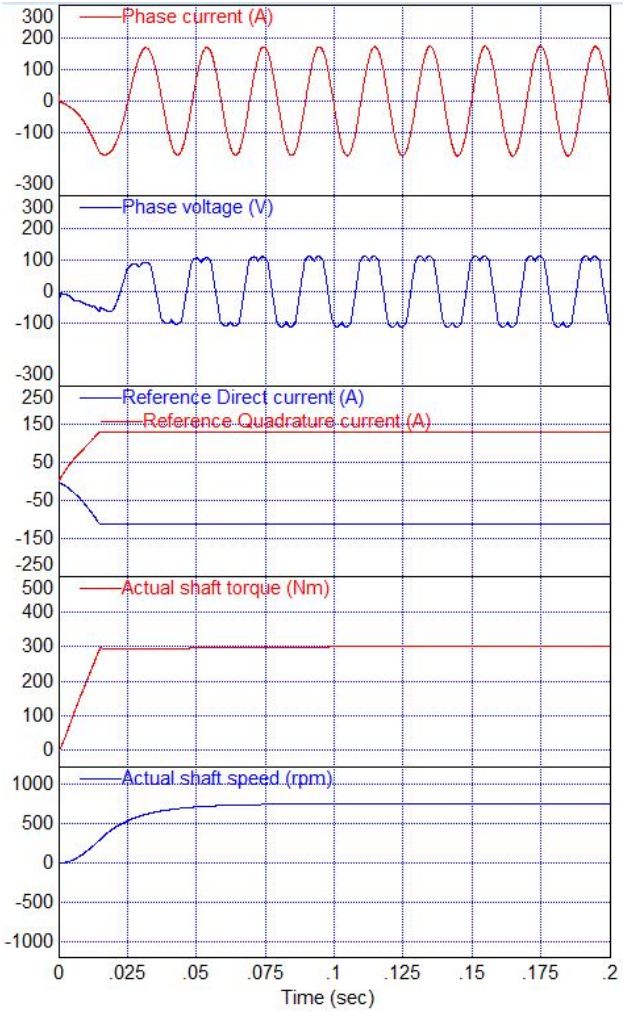


Fig. 9.24 SIM result, IPM operation, FOC-torque control active: mode SIM-saturation

9.1.1.3 Results, IPM sensored FOC with Enc: SIM-linear-saturation

This subsection considers the following options for the controller and IPM machine:

- Controller mode: `Linear Controller`
- IPM machine model: `saturation`

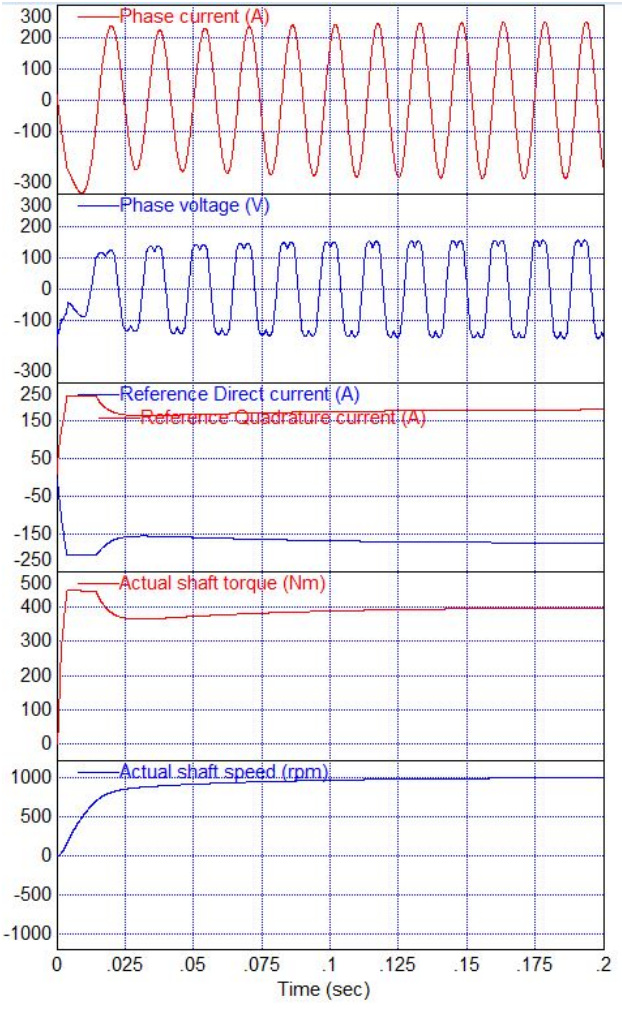


Fig. 9.25 SIM result, IPM operation, FOC-speed control active: mode SIM-saturation

Hence, operation where the controller is NOT matched to the IPM machine model, with 'deep' magnetic saturation and operation results for torque and speed control are presented, with the same setting as used in the previous two cases.

The results shown in figure 9.27, represents operation under torque control with T_{ref} =300 Nm, hence the steady state shaft speed is dependant on the applied load torque AND generated shaft torque. A comparison with results shown in figure 9.24 learns that the drive is NOT capable of reaching the required 300 Nm reference torque. This is due to the use of a linear controller with a 'deep saturated' IPM machine operating with torque control active.

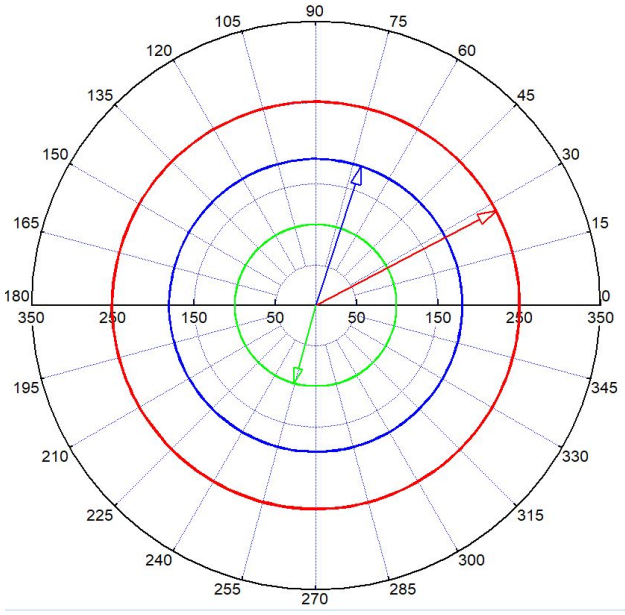


Fig. 9.26 SIM result, IPM operation, FOC-speed control active, vector plot: mode SIM-saturation

Figure 9.17, shown earlier, confirms these results as the direct/quadrature current amplitudes used in a 'linear' MTPA controller are significantly lower. An example with speed control active, as given in figure 9.28, shows that the speed controller current limit 300 A (set in the operation dialogue box) is initially reached and around that time mark 100 ms the reference speed (set to 1000 rpm) is reached. Correspondingly, the speed controller then provides the direct/quadrature currents (consistent with MTPA linear operation (in this example)) needed to generate a shaft torque that matches the load torque. A linear load torque/speed characteristic is assumed, as discussed earlier, which is set to produce 400 Nm at 1000 rpm. These results confirm that the required speed with matching shaft torque is provided by the drive. The important observation is that this is achieved with a 'linear' MTPA controller. Of importance is to consider the vector plot, see figure 9.29 at time mark: 0.2 s, which shows the orientation of the scaled (factor 100) unity d-axis ('green'), voltage vector ('blue') and the current vector ('red'). Observation of the current vector amplitude learns that it equal, when compared to the saturated case (see figure 9.26). It is also noted that orientation of the current vector relative to the controller d-axis has not significantly changed, when compared to the saturated case, this despite the fact that a linear MTPA controller was used in this example. This was expected given the results shown in figure 9.16, as mentioned above. This example shows the importance (when considering 'torque control' in particular) of matching the controller with IPM machines

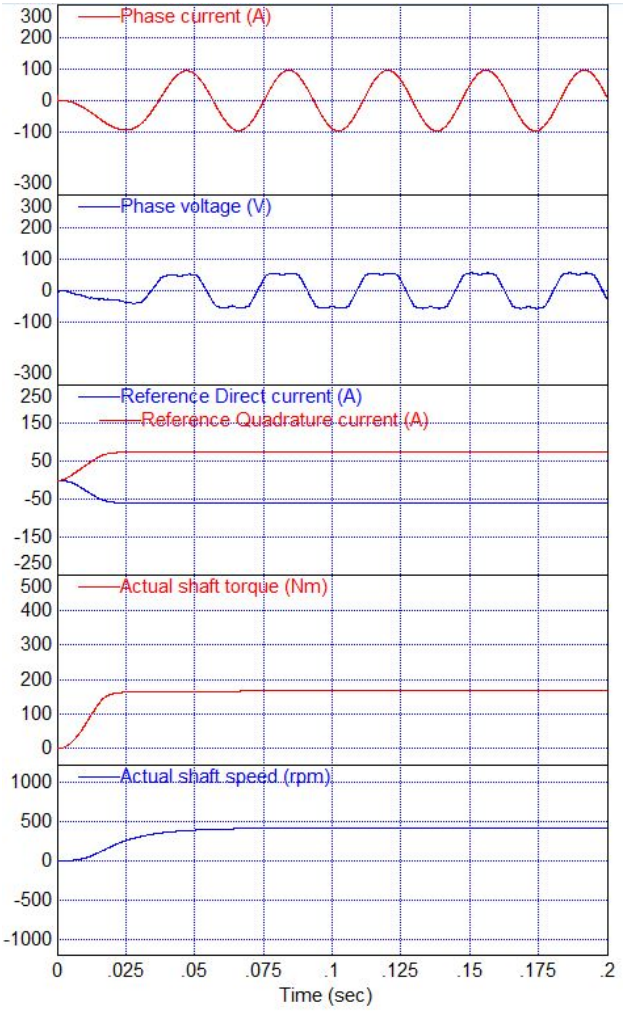


Fig. 9.27 SIM result, IPM operation, FOC-torque control active: mode SIM-linear-saturation

that have 'deep' saturation. Similar phenomenon also found for switched reluctance drives, which typically operate with 'deep' saturation, as will be discussed at a later stage.

9.1.2 IPM Sensored FOC with Enc: PIL

Reference files:

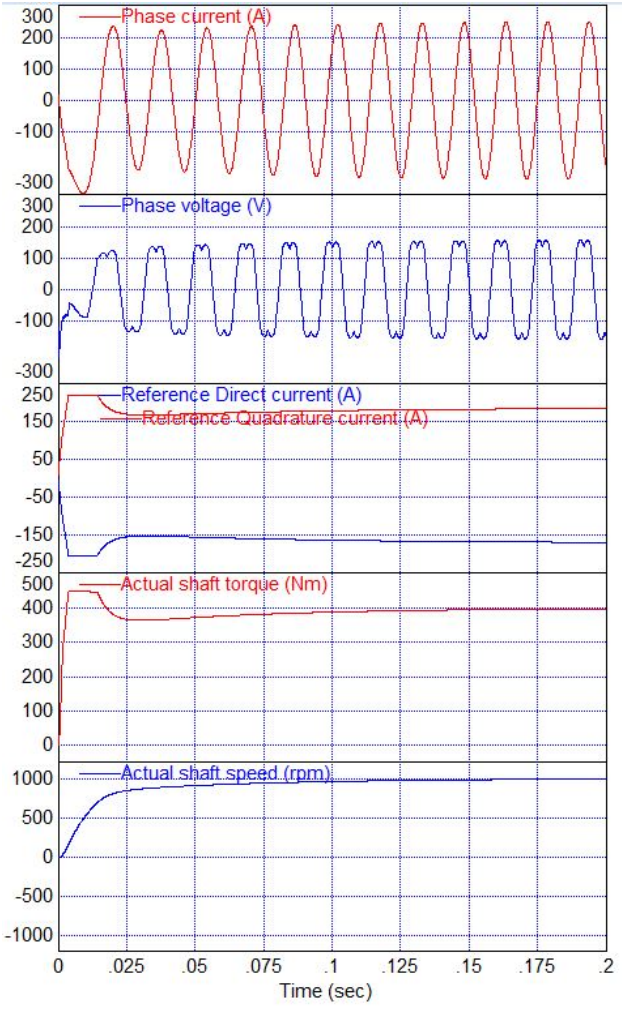


Fig. 9.28 SIM result, IPM operation, FOC-speed control active: mode SIM-linear-saturation

- `IPM-FOC-Lin-Sat-F069-PIL.vsm`
- `IPM-FOC-PIL_mod.vsm`
- `GA_inverse_dmap.csv`
- `GA_inverse_qmap.csv`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the fixed point controller within the target MCU, in this case the F28069M. This implies that the controller module shown in the simulation (see figure 9.19) is now replaced with a target execution module for file: `IPM-FOC-PIL_mod.out` as indicated in figure 9.30. An addition input

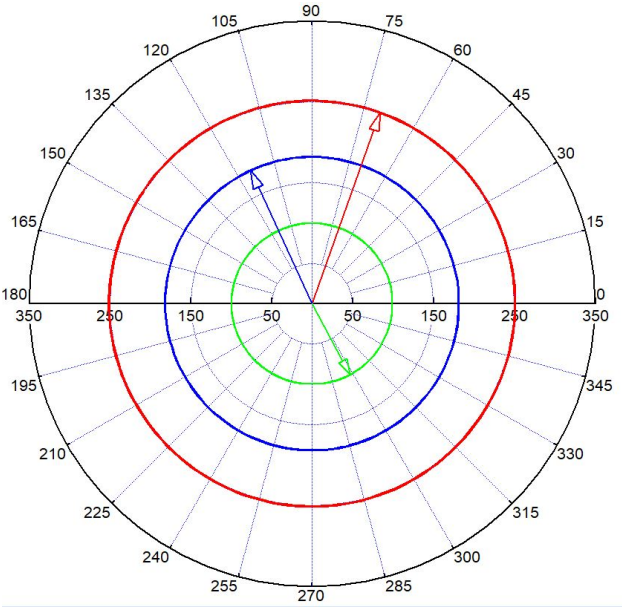


Fig. 9.29 SIM result, IPM operation, FOC-speed control active, vector plot: mode SIM-linear-saturation

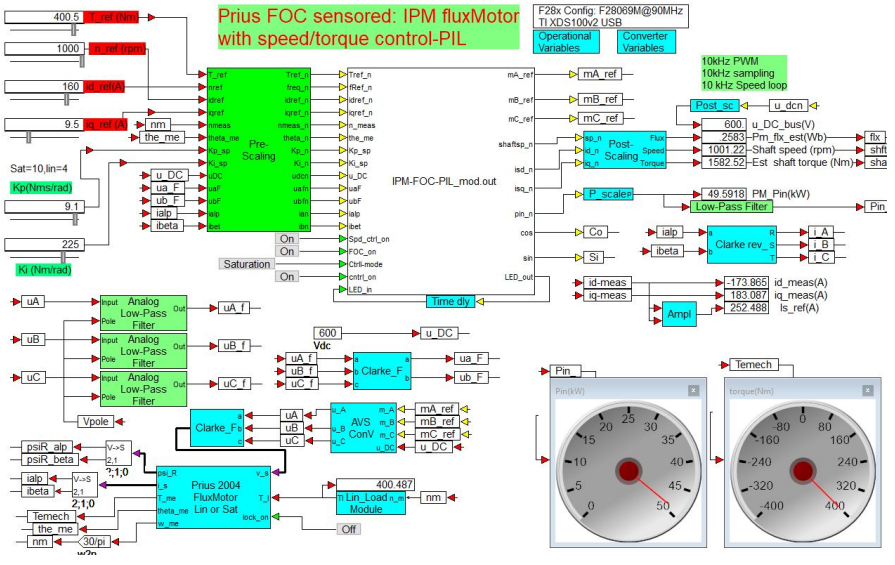


Fig. 9.30 Field-oriented torque/speed control with encoder:PIL operation

to the target interface module has been provided to accommodate the use of the fixed point controller for 'linear' or 'saturation' operation.
The target interface model that executes the out-file, generates the modulation indices required for the converter motor and these components are not affected by use of the PIL. Hence the FluxMotor IPM can be configured for operation with or without saturation using the dialogue box provided in the machine model. Generation of the out file is based on the PIL module, see figure 9.31, which must be compiled using the command: `toolbox>codegen`. Moving one

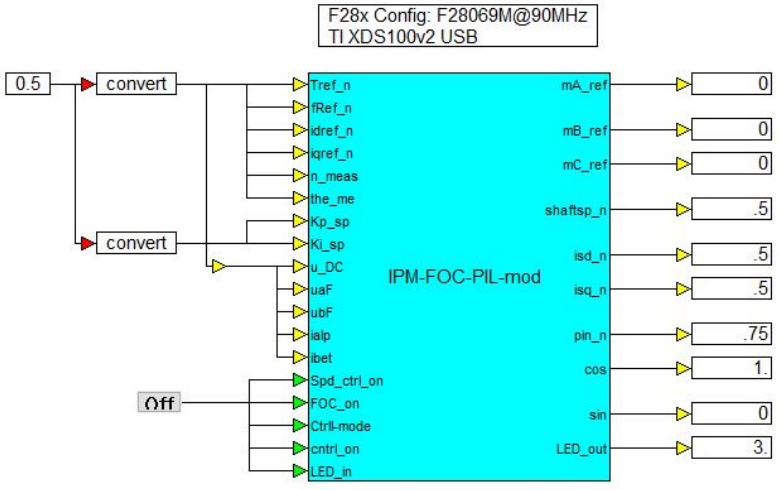


Fig. 9.31 PIL module, IPM FOC torque/speed control

level into the PIL module `IPM-FOC-PIL-mod` shows the 'IPM Drive controller' as discussed earlier (see figure 9.20). Added to this PIL module is a unit `LED_ctrl`, which is used to flash an LED on the MCU board to show that PIL control is functional.

Running the PIL file by starting the file ('green arrow' when using figure 9.30) will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational. Results produced by PIL operation (running in the example shown with FOC-speed control active and saturation option active for motor and controller) are identical to the plots shown in the 'simulation' (SIM) section (see section 9.1.1) and are therefore not shown here. The value of PIL operation is to verify operation of the controller on the MCU prior to, for example, moving to HIL. Furthermore, PIL operation is valuable for analysis of sensorless control, where use is made of, for example, FAST®(located in the F28069 MCU).

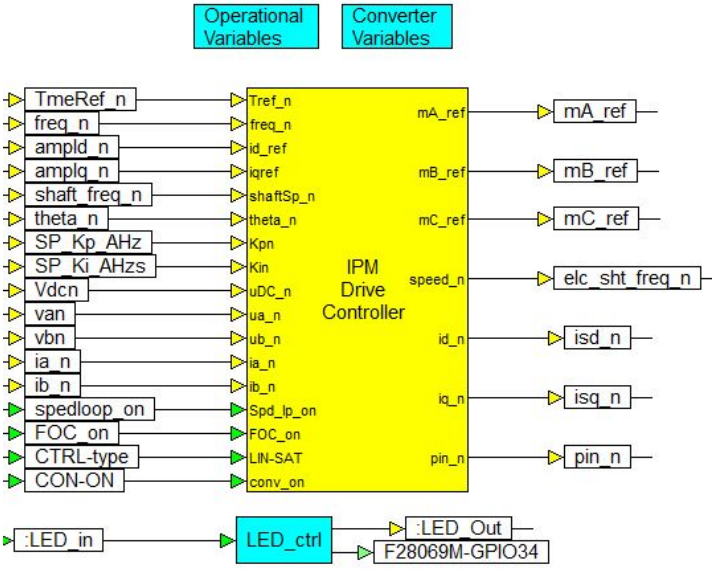


Fig. 9.32 Content PIL module, Field-oriented torque/speed control FluxMotor IPM with encoder

9.1.3 IPM Sensored FOC with Enc: COSIM

Reference files:

- `IPM-FOC-FluxM-COSIM-sat.vsm`
- `PriusConvSAT.psimsch`
- `IPM-Sat_v3.psimsch`

To evaluate operation of a controller with a generic switching converter, use can be made of a `PSIMCoupler` module, as indicated in figure 9.33. This approach implies the use of PSIM together with Embed, which is referred to as 'co-simulation'. As such, this approach provides the ability to evaluate the performance of a switching converter connected to an IPM motor with (in this example) magnetic saturation. In addition, torque ripple effects, due to the use of a switching converter can be examined using the COSIM approach. For this example, use is again made of the flux-linkage/current characteristics generated with the aid of a FluxMotor [19] finite-element software, as

discussed earlier (see the introduction of this chapter).

The `Drive Controller`, hatched in this case as its sampling frequency is

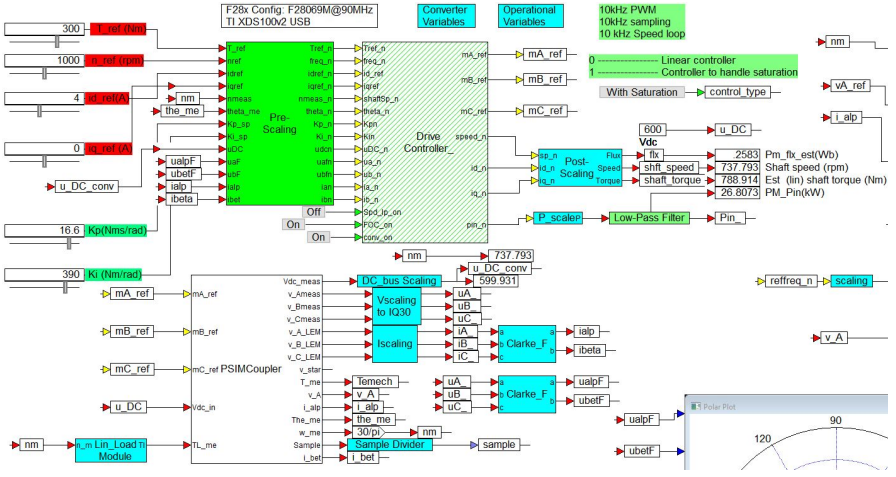


Fig. 9.33 IPM-PSIMcosim based drive, field-oriented torque/speed control with encoder

locally set to 10 kHz, whereas the 'system' frequency (set in the 'system properties' dialogue box) is set to 20 MHz. The latter is much higher, given the need to accurately represent 10 kHz voltage switching waveforms generated by the PWM unit of the converter. The content of the `Drive Controller` module was introduced earlier, see figure 9.20. However, the current and voltage measurements now make use of the signal processing circuitry connected to the half-bridges.

In this case the current transducers are in the converter phases, as shown in figure 9.34. After sampling and signal-processing in the converter, these variables appear as: `i_A`, `i_B`, `i_C` in figure 9.33. The low pass filtered voltages (which removes the PWM component using R-C filters, see figure 9.34) of the three half-bridges are also present in figure 9.33 and are also scaled and sampled by the module `Vscaling to IQ30`. In this case the per unit variables are rescaled to actual values: `u_A`, `u_B` and `u_C` respectively.

Further observation of figure 9.34 shows the presence of a module `IPM`, which is a generic IRTF model of the FluxMotor [19] IPM synchronous machine machine, WITH (in this case) saturation, as presented earlier (see figure 2.18). However, the PSIM non-linear IRTF IPM model, shown in figure 9.35, has an 'electrical interface' to allow use of said model with the converter.

Also shown in the PSIM IPM model are two look-up tables referred to as `idtest2.tbl` and `iqtest2.tbl` respectively. These are the inverse flux-linkage/current characteristics of the FluxMotor IPM as discussed earlier, see

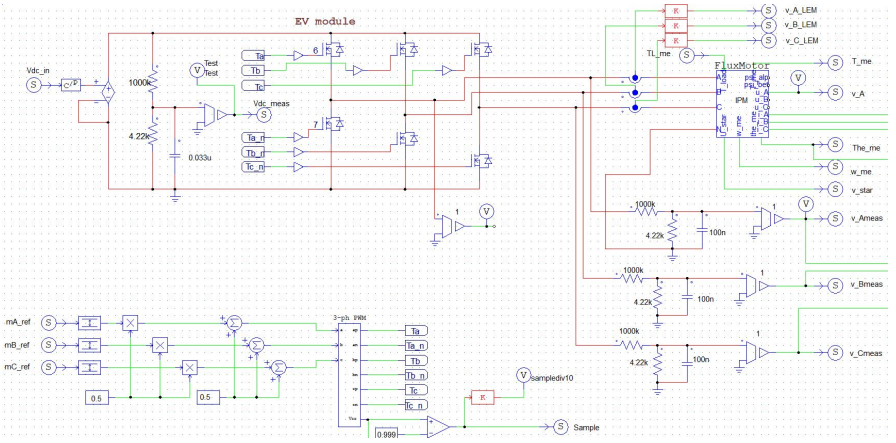


Fig. 9.34 Content PSIM coupler module, generic converter with FluxMotor IPM (with saturation)

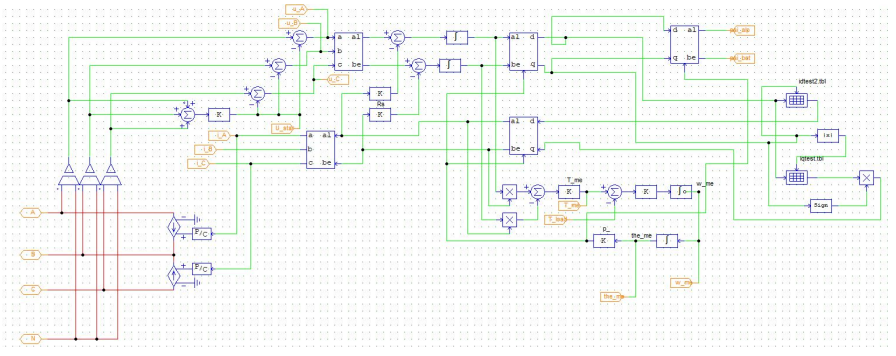


Fig. 9.35 PSIM sub-module: 'IPM'

figure 9.15.
A PWM unit referred to as `3-ph PWM` (see figure 9.34), generates the gate control signals used to switch on/off the upper or lower devices of each converter leg.

Central to sensorless operation is the ability to accurately measure the phase voltages and currents. The experimental drive board(s) used in this GUIDE are mostly 'sensorless prepared', which implies that they have been purposely designed with a data acquisition module (ADC) that undertakes the signal conditioning of the voltage/currents as needed to realize sensorless (no mechanical encoder attached) control. In addition, access to voltages and currents, gives the user ability to estimate the motor flux and input power. The latter can be used to estimate energy usage Wh, which is convenient for battery operated drives.

The converter model shown here is 'sensorless prepared' and an example of encoderless operation using the FluxMotor IPM will be discussed at a later stage in the GUIDE.

The 'Converter Variables' dialogue box, given in figure 9.36, defines (among others) the scaling used by the Analog to Digital converter, which has a maximum input value of 3.3 V and is directly linked to the signal processing undertaken for voltages and currents as discussed above. For example, the

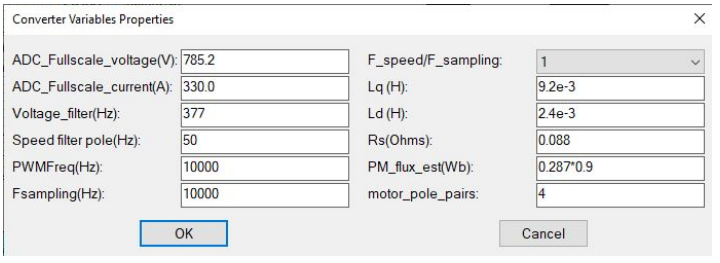


Fig. 9.36 Dialogue box: 'Converter Variables'

'ADC-fullscale-voltage' value is the theoretical peak-to peak input voltage which must be applied to the low-pass filter network (see figure 9.34) to ensure that the ADC input level varies between $0 \rightarrow 3.3$V. The current transducer voltage(s) `v_A_LEM`; are sampled by the `Iscaling` module shown in figure 9.34. When dealing with the actual MCU, the ADC converter also takes over this sampling process. Apart from the measured voltage/currents generated by the converter, the PSIM coupler unit also provides the actual motor model current vector:`i_alp`, `i_bet` (in stationary coordinates), phase 'A' voltage and speed, which are presented by the variables `v_A`, `i_alp`,`i_bet` and `n_m` (after conversion to rpm) respectively, see figure 9.34.
Added to the 'Converter Variables' dialogue box (see figure 9.36) are the linear IPM motor parameters, as the controller can operate with and without the effects of magnetic saturation.

9.1.3.1 Results, IPM sensored FOC with Encoder

In this section the results are shown for the COSIM example introduced above. Two modes of drive operation: 'torque control' and 'speed control' will be shown with the IRTF FluxMotor model, that is operating with magnetic saturation. The controller has the option of 'linear' or 'saturation' mode, which implies that the MTPA characteristic is adapted accordingly, as discussed earlier (see section 9.1.1).

Operation under speed control, with a speed reference setting of 1000 rpm is shown in figure 9.37. Indicated in said figure is the phase 'A' voltage waveform

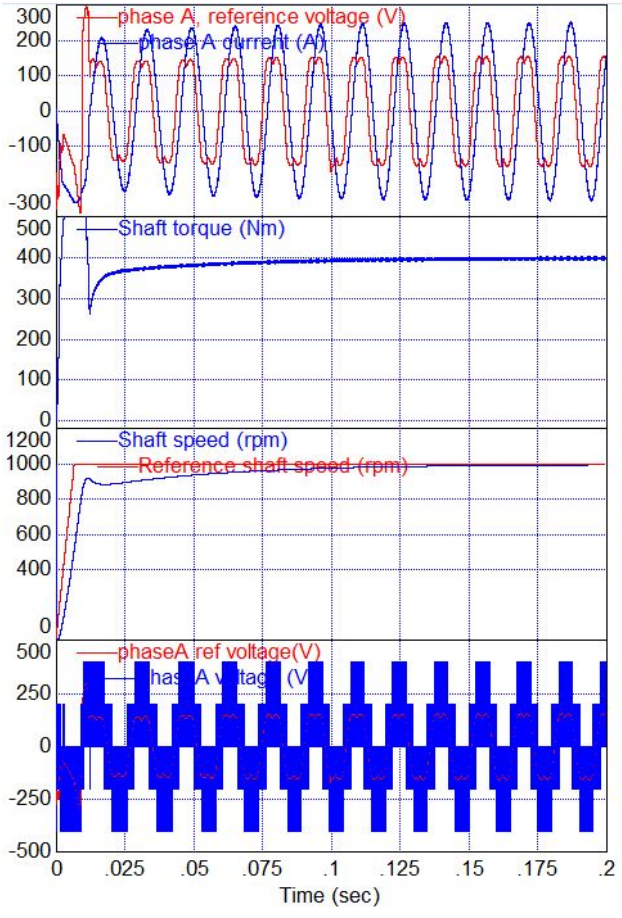


Fig. 9.37 COSIM result, FOC-speed control of the FluxMotor IPM with encoder

together with the reference voltage, phase current, shaft torque and shaft speed. By way of illustration the controller mode is initially set to 'linear' and time mark 0.1 s is switched to 'saturation'. The aim being to examine the effect of a controller change on the time waveforms over the period $0 \rightarrow 0.2$s. A linear load characteristic is again used which provides 400 Nm at 1000 rpm. Observation of the results shows that the controller mode switch does NOT significantly affect drive operation. This is expected as mentioned earlier (see section 9.1.1.3). The impact of converter switching is present on the shaft torque waveform, but torque ripple is very limited.

A vector plot at time mark 0.2 s , with operation under speed control and controller mode 'saturation' is given in figure 9.38. Shown in this figure

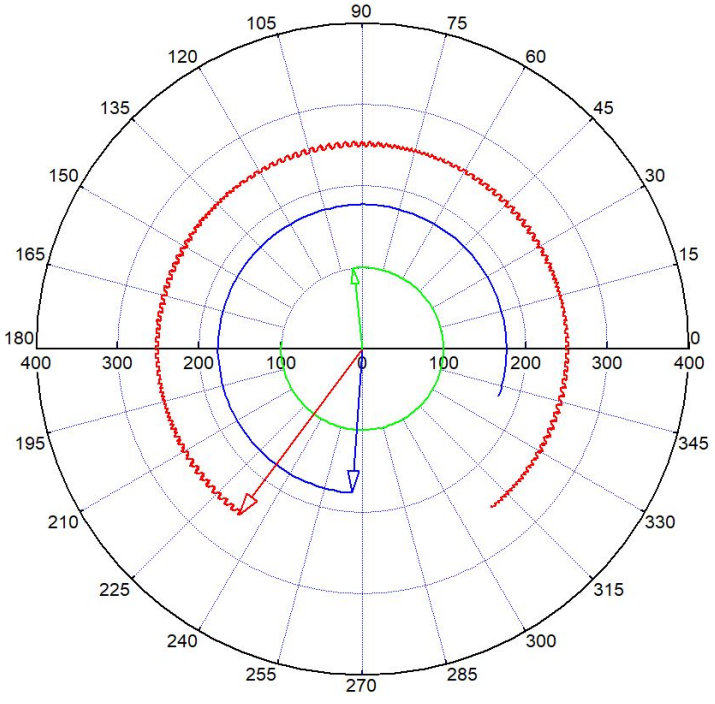


Fig. 9.38 COSIM result, IPM operation, FOC-speed control, vector plot: controller mode: saturation

are the measured LPF voltage vector ('blue'), a scaled unity d-axis vector ('green') and the actual (from the IRTF motor model) current vector ('red'). Note that the impact of switching is reflected in the current vector locus, e.g. changes in end-point direction due to use of 'active' or 'zero' voltage vectors [13]. A similar vector plot, under the same operating conditions, but with a non-switching (AVS) converter, as given in figure 9.26 underlines the importance of COSIM operation, i.e to observe the impact of using a switching converter on the torque and currents.

Operation under torque control with the controller mode initially set to 'linear' and at time mark 0.1 s switched to 'saturation' is shown in figure 9.39. Note that the 'Reference shaft speed' shown in figure 9.39 is not relevant under torque control as the load characteristic determines the steady-state shaft speed. The torque reference is set to 300 Nm and observation of the results show that the shaft torque of the IPM (with saturation) does NOT match the reference when using the 'linear' MTPA controller mode. However,

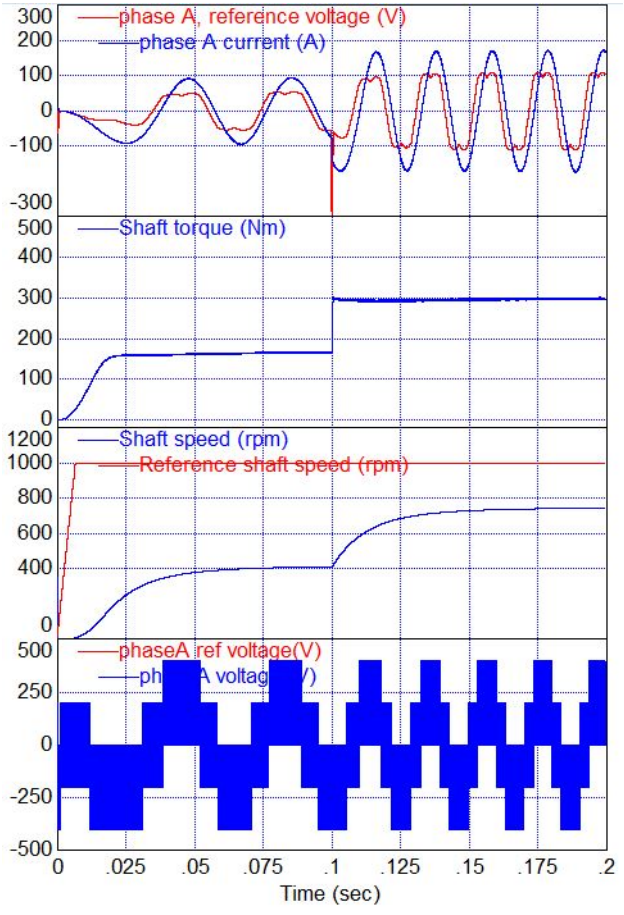


Fig. 9.39 COSIM result, FOC-torque control of the FluxMotor IPM with encoder

at time mark 0.1 s the controller is switched to MTPA 'saturation' mode. This implies that the controller mode is then matched to the machine and reference torque matches the machine shaft torque.
This type of analysis again confirms the importance of matching the controller MTPA to the machine and this is particularly relevant when 'deep magnetic saturation' is present, as is the case for this FluxMotor example.

9.2 IPM Sensorless Field Oriented Control

The term 'field-oriented control' (FOC) implies that a synchronous 'd,q' coordinate system is tied to the PM flux vector of the IPM machine (see

section 2.1.3.3). More specifically, the 'd-axis' of said coordinate system is coincidental with the flux vector $\vec{\psi}$, as may be observed from figure 9.40. To achieve the best possible torque for the lowest current vector amplitude (referred to as the Maximum Torque Per Ampere (MTPA) operating mode), use must be made of a controller (as discussed in the IPM introduction) that identifies the appropriate orientation of the current vector in the 2nd quadrant (in this case, because assuming a 'flux-weakening' IPM topology) of the d, q coordinate system.

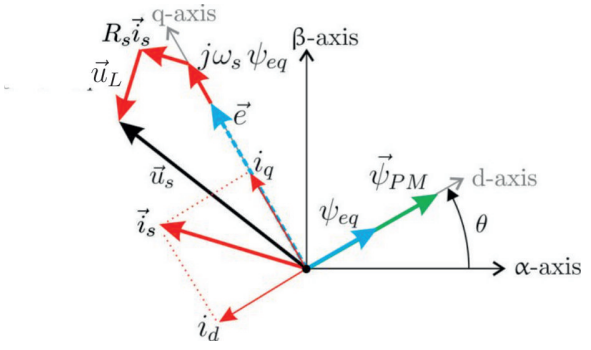


Fig. 9.40 Vector diagram: sensorless field-oriented control (FOC)

The term 'sensorless' FOC, implies the omission of a mechanical sensor typically used to align the controller d-axis with the magnetic flux vector. For 'encoderless' (sensorless) field oriented control use is made of an algorithm, such as FAST®, to identify the magnetic flux vector (amplitude ψ_{PM} and orientation θ in α, β coordinates) of the IPM machine. Correspondingly, the 'd-axis' is then aligned with the estimated magnetic flux vector of the IPM machine.
For IPM machines the process of flux estimation is also (as done for IM and PMsyn) tied to EMF identification, which is based on measurement of the terminal voltage vector $\vec{u}_s$ and current vector $\vec{i}_s$. This EMF identification can be done in two ways (as will be discussed with the aid of figure 9.40) namely:

- Option 1: Use of the voltage $\vec{u}_L$ across the quadrature inductance L_q.
  This approach identifies a resultant EMF due to PM flux AND the equivalent flux $\psi_{\text{eq}} = i_d \Delta L$ hence, $\vec{e}_{\text{tot}} = \vec{e} + \vec{e}_{\text{eq}}$ (which are both aligned with the 'q-axis') with $\vec{e}_{\text{eq}} = j\omega_s \psi_{\text{eq}}$. Identification of the EMF vector $\vec{e}_{\text{tot}}$ is then carried out using:

$$\vec{e}_{\text{tot}} = \vec{u}_s - \vec{u}_L - R_s \vec{i}_s \tag{9.8}$$

The advantage of this approach is that it requires knowledge of the voltage across the stator resistance and (only) the quadrature axis inductance defined as: $\vec{u}_L = j\omega_s L_q \vec{i}_s = j\left(\vec{i}_s/i_{\rm sc}\right) e$, where $i_{\rm sc} = \psi_{\rm PM}/L_q$ is the short-circuit current of the IPM machine. Furthermore, torque estimation is simplified, as a sensorless algorithm (that uses this option) identifies the resultant flux $\psi_{\rm tot} = \psi_{\rm PM} + \psi_{\rm eq}$, which means that a shaft torque estimate can be found using $T_{\rm me} = {}^3\!/_2\, p\, i_q\, \psi_{tot}$. For machines where saturation cannot be ignored the quadrature inductance becomes a function of the stator current amplitude. This is particularly important for machines (like the FluxMotor IPM), which typically operate with stator current amplitudes in excess of the short-circuit current (hence the amplitude of voltage $|\vec{u}_L|$ is greater then the EMF $|\vec{e}|$).

- Option 2: Use of the voltage across the quadrature inductance L_q AND the incremental inductance $\Delta L = L_d - L_q$.
  This approach identifies the EMF due to PM flux hence, $\vec{e}$ (which is aligned with the 'q-axis') with $\vec{e} = j\omega_s \psi_{\rm PM}$. Identification of the EMF vector $\vec{e}$ is then carried out using:

$$\vec{e} = \vec{u}_s - \vec{u}_L - R_s \vec{i}_s - j\omega_s \psi_{\rm eq} \tag{9.9}$$

This approach requires knowledge of the voltage across the stator resistance, the quadrature axis inductance $\vec{u}_L = j\omega_s L_q \vec{i}_s$ and the voltage across the incremental inductance $j\omega_s \psi_{\rm eq}$. Hence, this approach requires knowledge of the direct and quadrature inductances. Furthermore, torque estimation is more complex, as a sensorless algorithm (that uses this option) identifies the PM flux $\psi_{\rm PM}$, which means that a shaft torque estimate must now be found using $T_{\rm me} = {}^3\!/_2\, p\, i_q\, (\psi_{PM} + i_d \Delta L)$. For machines where saturation cannot be ignored, both direct and quadrature inductances need to be a function of the stator current amplitude when using this option for sensorless EMF estimation.

Note that for synchronous reluctance drives (to be discussed at a later stage) only option '1' is usable (no PM flux present). Furthermore, when saturation of an IPM is not serious, use can be made of the linear MTPA algorithm (based on the use of expression (9.5)), which does require knowledge of the PM flux amplitude and the incremental inductance.

For this reason field-oriented control examples often include open loop current & frequency control as a method to check alignment of the controller 'd-axis' with the PM flux vector (with the sensorless algorithm active), when operating

open loop (under mechanical no-load conditions) and identification of IPM machine parameters such as PM flux ψ_{PM} and incremental inductance ΔL, as will be discussed below.

The tool chain as introduced in figure 2.45 is partially used as a template for the ensuing subsections, here the toolchain: PIL→COSIM, will be considered, where initially use is made of an 'ISEA IPM' [1] machine, which can be practically deployed with a magnetically linear controller. At a later stage use is made of the IRTF IPM FluxMotor [19] example introduced earlier, which is configured for a 'deep magnetic saturation' IRTF model and a sensorless controller, which can be configured for MTPA operation using control mode: linear or saturation.

9.2.1 ISEA IPM sensorless FOC using FAST: PIL

Reference files:

- `FOC_IPM_FAST_F069-ISEA-PILn3.vsm`
- `DP-IPM-ISEA-FAST-PIL_mod.vsm`
- `ISEA-IPM-FOC-limit-calculator.vsm`

In this section sensorless FOC operation of a so called 'ISEA IPM' motor [1], as used by RWTH-ISEA, Germany in their electrical drive laboratory. Saturation levels of this machine are not significant, hence a linear IPM model and MTPA controller can be used. A Texas Instruments `8301-HC` converter with `F28069M` ISO control card that has the FAST®sensorless algorithm, is used in this example as shown in figure 9.41.
The Processor In the Loop (PIL) approach discussed here, is designed to evaluate the controller within the target MCU, which contains the DLL used to communicate with FAST®, as mentioned above. This implies the use a target execution module for file: `IPM-FOC_FAST-PIL.out` as indicated in figure 9.42.
Generation of the 'xxx.out' file is based on the PIL module, see figure 9.43, which must be compiled using the command: `toolbox>codegen`. Moving one level into the PIL module `IPM-FOC-FAST-PIL` shows a 'Drive Controller', see figure 9.44. Added to said controller is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that PIL control is operational.
The content of the 'Drive Controller' as given in figure 9.45, shows the `Current controller` and `Speed Controller` as introduced earlier, see section 4.3.1.1.
However, the per unit inverse bus voltage variable `invVdc` is now generated by the FAST®module `InstaSPIN FOC`. Said unit provides per unit estimates of the rotor flux amplitude `ROTORFLX`, rotor flux angle `ang` and electrical

(a) Converter-F28069 MCU

(b) ISEA IPM flux-weakening

Fig. 9.41 ISEA-IPM drive set-up for sensorless control

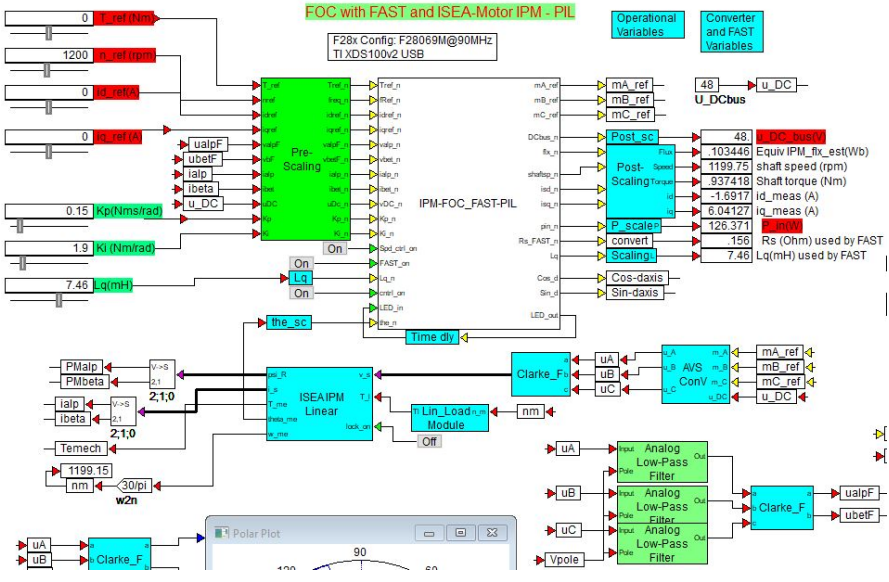


Fig. 9.42 Field-oriented MTPA torque/speed control for ISEA-IPM using FAST®: PIL operation

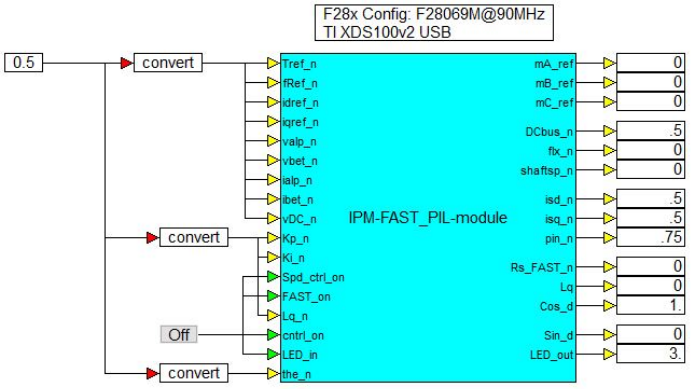


Fig. 9.43 Module:'IPM-FOC-FAST-PIL'

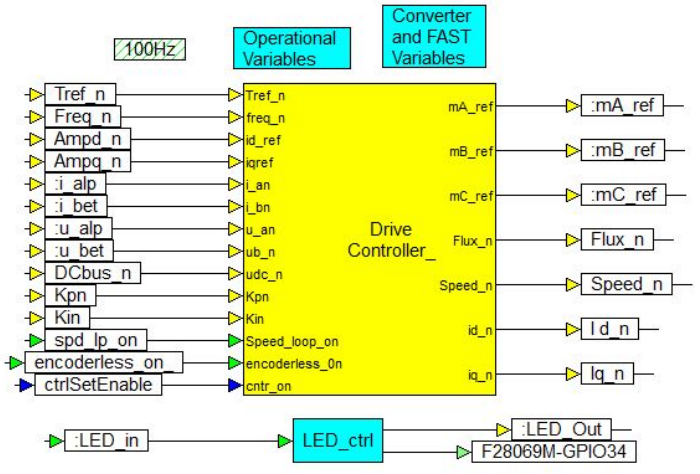


Fig. 9.44 Content PIL module, Field-oriented torque/speed control with FAST®

shaft frequency `shaft_freq_n`. Inputs to the FAST®module are the per unit measured LPF voltages and currents which are Clarke converted to stationary α, β coordinates. In addition, a variable `CurCTLstate` (generated in the `100 Hz` module) activates the FAST®module, which is also provided with the per unit measured DC bus voltage `Vdcn`. Also shown in figure 9.45 is a module `IPM-Is Controller Lin module`, as discussed earlier (see figure 9.4) which, in this case, is provided with ISEA IPM parameters. Outputs of the this unit are the MTPA per unit direct /quadrature current reference values, active when 'speed control' is selected. Also shown in this figure is the module `Lin-MTPA` (as introduced earlier, see figure 9.6), which has as input the per unit torque reference and output the MTPA optimized per unit direct/quadrature

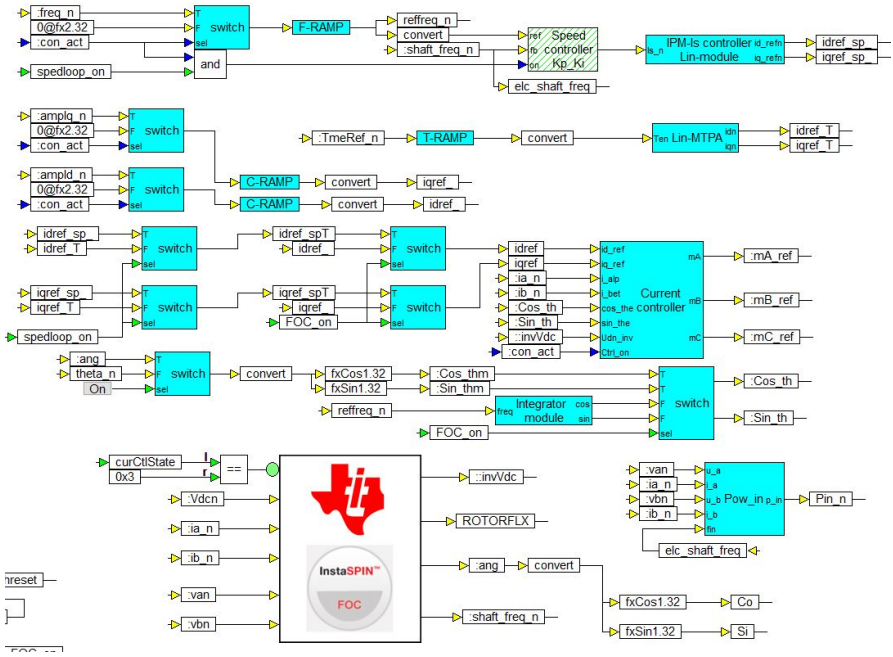


Fig. 9.45 Content IPM 'Drive Controller' module

currents, active when 'torque control' is selected.
When FAST® is not used (encoderless off) the logic variable `FOC_on` is zero and this implies open loop current&frequency control is active. However, the FAST® module will also remain active under these conditions, hence an effective mode to test operation of this module.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational.

Operation of the drive is with a speed reference 1200 rpm and the load controls the vector current amplitude (using the speed controller once steady state operation is achieved). For these results (see figure 9.46) a linear load torque-speed characteristic is assumed, which generates a torque of 1.5 Nm at 2000 rpm. FAST® operation starts at 0.32 s and the torque is initially limited by the 15 A vector current amplitude setting of the speed controller.

The vector plot given in figure 9.47, which corresponds to time mark 0.5 s, is interesting, as it shows the scaled (by a factor 200) PM flux vector ('green'), unity 'd-axis'('brown' and scaled by a factor 200), current vector ('red') and terminal voltage vector ('blue'). A key observation from this vector plot is the orientation of the current flux vector relative to the rotor flux vector. For

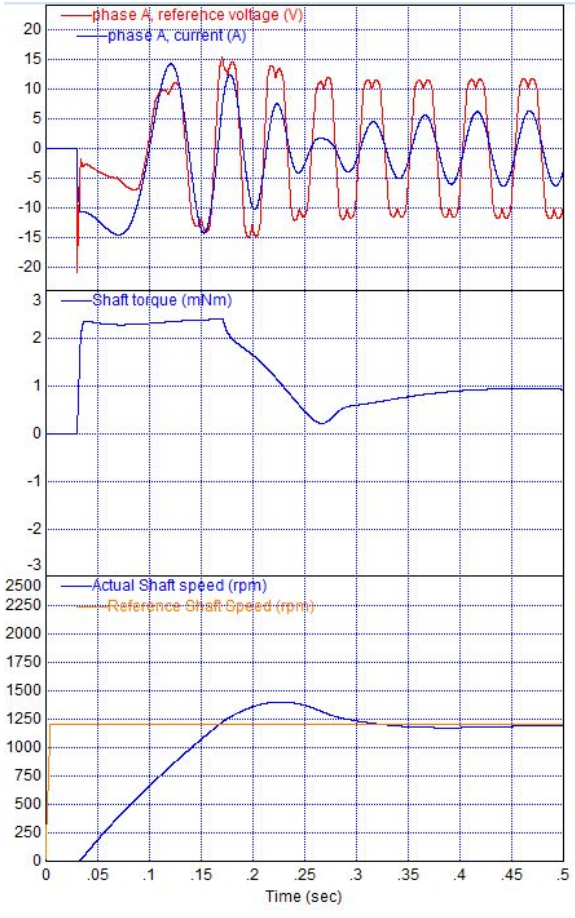


Fig. 9.46 PIL Result: FOC MTPA sensorless operation ISEA-IPM drive, using FAST®with speed control

MTPA operation at the steady-state operating point the direct and quadrature currents are equal to -1.7 A and 6.0 A respectively, as may be observed from figure 9.47. Also note that the 'd-axis' is indeed aligned with the PM flux vector, which implies that the 'Lq' slider setting in figure 9.42, matches that of the machine. Furthermore, the sensorless algorithm flux estimate (also shown in figure 9.42) is equal to 0.103 Wb, which is the sum of the equivalent flux ψ_{eq} and the PM flux ψ_{PM} equal (for this machine) 95 mWb. Hence, the equivalent flux is equal to 8 mWb, that must be equal to the $\psi_{\text{eq}} = i_d \, \Delta L$, as is indeed the case.

Torque control is active when button `FAST_on` is `ON` and `Spd_Ctrl_on` is `OFF` (see figure 9.42). MTPA FOC operation of the drive with a torque refer-

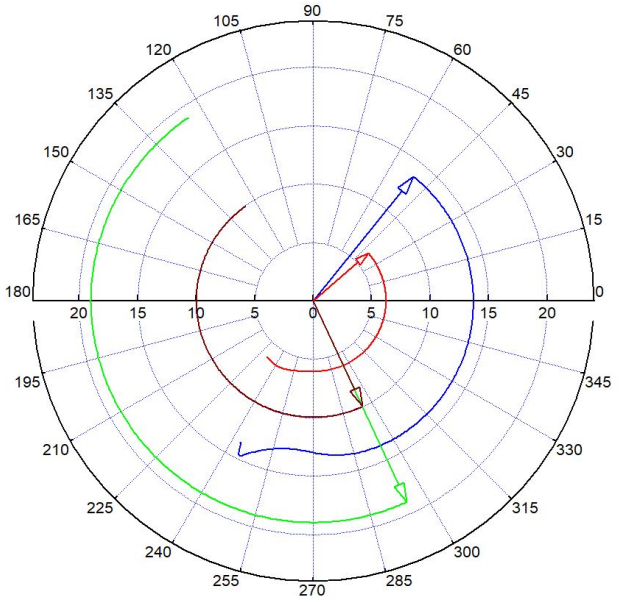


Fig. 9.47 PIL result, IPM drive operation, MTPA sensorless FOC-speed control active:vector plot

ence value of 1.0 Nm is shown in figure 9.48.
In this case the load-torque characteristic (as mentioned above) determines the steady-state speed.

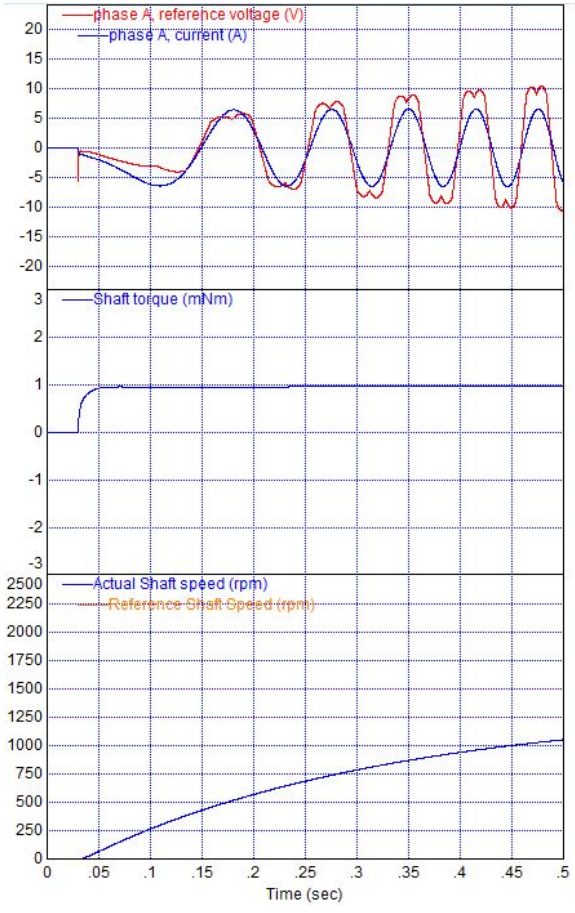


Fig. 9.48 Results ISEA-IPM: sensorless FOC torque control, MTPA operation

The third reference file: `ISEA-IPM-FOC-limit-calculator.vsm` mentioned in the introduction of this section, provides a user with the ability to determine the limits of MTPA FOC operation of an specific drive, in this case the 'ISEA IPM' operating with a FOC converter connected to a DC bus. Inputs to this 'FOC limit estimator', shown in figure 9.49 are the drive: DC bus voltage `u_DC(V)`, maximum modulation index (used by the current controller) `max mod index` and current vector amplitude `is_ref)A)` as used by the speed/torque controller to set the direct/quadrature axis current references. The dialogue box, also given in this figure, represent the parameters of the ISEA IPM. Outputs, which appear (after running this file) are (among others)

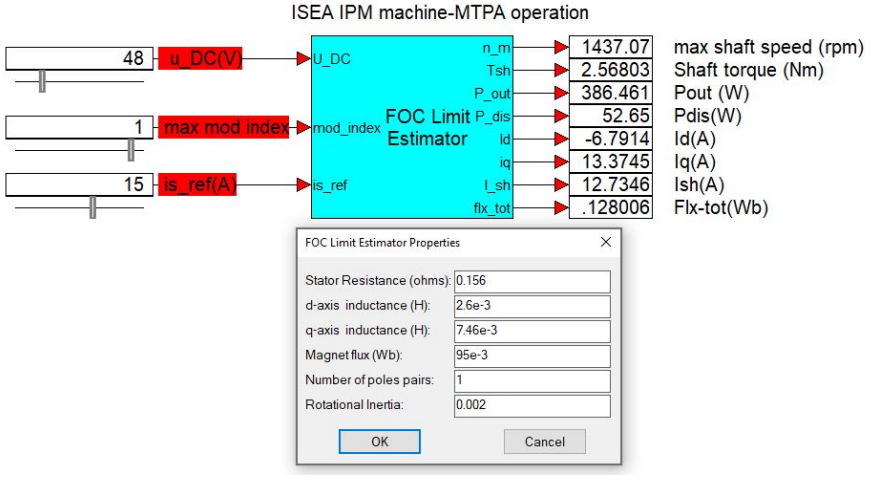


Fig. 9.49 Results ISEA-IPM: FOC Limit Estimator, MTPA operation

'max shaft speed', which is highest speed possible with FOC (and operation with $i_q^{\text{ref}} = i_q^{\text{meas}}$). This limit is reached at a speed when the current controller modulation index reaches the user defined value. When operating above this speed the quadrature current controller starts to operate under voltage control, hence the measured i_q current will reduce, which also reduces the torque as operation moves away from the MTPA operating point (as defined in this estimator by the slider `Is_ref`). Other outputs are: shaft torque, output/dissipated power, direct/quadrature axis currents (that correspond to MTPA condition), short-circuit current and the total flux (the sum of the PM flux and equivalent flux). This is convenient as it allows a calculation of the torque contribution due to the use of reluctance (and introducing a negative direct axis current component according to MTPA operation).
For example, with the settings and output variables shown in figure 9.49 the electrodynamic torque can be calculated using $T_{\text{me}}^{\text{eldyn}} = {}^3\!/_2\, \psi_{\text{PM}}\, i_q$ is equal to

1.9 Nm , hence a reluctance torque component of 35 % (relative to the PM electrodynamic torque) is present.
Its also important to note that the short-circuit current of this IPM is in the same order as the quadrature current (given the input current setting shown), hence the voltage across the quadrature inductance is of the same order as the EMF (due to the PM). Consequently, when operating with sensorless control (as discussed in the ensuing section), it important to have an accurate estimate for L_q (otherwise phase errors between the controller d-axis and PM flux vector will occur). This issue is of particular importance when dealing with 'deep saturation' IPM's, like the FluxMotor example discussed earlier (for sensored FOC).

9.2.2 FluxMotor-IPM sensorless FOC using FAST®: PIL-COSIM

Reference files:

- `IPM-PIL-FluxM-COSIM-sat.vsm`
- `DP-IPM-FluxMotor-FAST-PIL_mod.vsm`
- `IPM-Sat_v3.psimsch`
- `PriusConvSAT.psimsch`
- `param_file.txt`

This example considers the simulation of a IPM FluxMotor [19] connected to an generic switching converter, introduced earlier (see figure 9.34), which is in turn connected to a 'IPM FluxMotor FAST-PIL' module, as shown in figure 9.50. The aim is to examine sensorless (encoderless) FOC operation of an IPM drive using a fixed point controller with FAST®. More specifically, the FluxMotor [19] IPM (as discussed earlier, see section 9.1.1) is again used here, to illustrate the impact (on sensorless drive control) of using a machine with 'deep magnetic saturation'. This machine is represented by a IRTF based 'IPM' module, which makes use of direct/quadrature axis inverse flux-current characteristics (introduced earlier, see figure 9.11) generated with the aid of FluxMotor® [19] finite-element software. This module, introduced earlier (see figure 9.35), is located inside a PSIM coupler module and is used again in this IPM encoderless drive example (see figure 9.50).

Inputs to the controller are the direct/quadrature axis reference currents `id_ref`,`iq_ref`, shaft speed reference `n_ref` and shaft torque reference `T_ref(Nm)`. In addition, the speed controller proportional `Kp` and integral gain `Ki` inputs (used when speed control is enable) are also provided. Furthermore, an input `the_n` is provided, which is a per unit electrical shaft angle that can be used for debugging purposes. This implies that the PIL module can internally be switched to sensored FOC , if needed to examine/debug

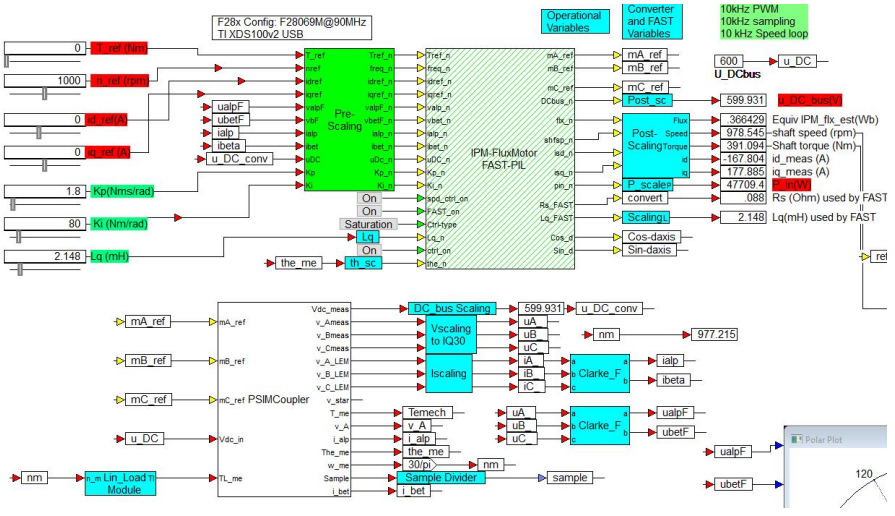


Fig. 9.50 Encoderless Field-oriented torque/speed control FluxMotor IPM: PIL-COSIM

sensorless control issues, as will be discussed later.

Four logic inputs are shown that provide the following functionality:

- `Ctrl`: converter on/off control.
- `FAST_on`: Field-oriented control active using FAST®.
- `Spd_ctrl_on`: Operation under ' FOC speed control' when button ON and 'FOC torque control' with button OFF.
- `Cntrl-type`: which allows the user to select one of two options namely:
  - `Linear`: which sets controller operation based on a linear model of the FluxMotor IPM.
  - `Saturation`: which sets controller operation based on a FluxMotor model with magnetic saturation.

When the logic inputs `FAST_on` and `Spd_ctrl_on` are both OFF, drive control is C&F and the direct/quadrature sliders control drive operation together with the shaft speed reference slider.
Note that when operating under torque control, the direct/quadrature axis current references are chosen to optimize torque production, i.e MTPA operation as discussed in the introduction. This basically means that the direct/current reference values as function of the shaft torque are set by the 'linear' plots shown in figure 9.17 (which were generated for the FluxMotor example used here) when option 'linear' is selected. Vice-versa when option 'saturation' is selected, the plots 'sat-case curve fit', also shown in figure 9.17 are used by the controller.
When operating under speed control, the speed controller generates a vector

current amplitude reference, which is converted in the controller to optimum MTPA direct/quadrature reference values, as shown in figure 9.16 (which are tailored to the FluxMotor example). Pending the choice of controller (option 'linear' or 'saturation') the direct/quadrature current plots as function of current amplitude are selected appropriately. The aim of the speed controller is to identify the required current vector amplitude to meet the criteria: measured shaft speed equals user defined reference speed.

Outputs of the controller are (among others) three modulation indices, that are used to control the three converter half bridges inside the 'PSIMCoupler' module. Furthermore, the following outputs (after scaling) are provided:

- `u_DC_bus`: measured DC bus voltage.
- `Equiv IPM flx_est`: total flux due to the PM and the term $i_d \Delta L$ (use of 'option 1', see equation 9.9).
- `shaft speed`: estimated speed generated by the FAST®module.
- `shaft torque`: estimated shaft torque calculated using the total flux and quadrature current module (by use of 'option 1', see equation 9.9).
- `id_meas`: measured direct axis current.
- `iq_meas`: measured quadrature axis current.
- `P_in`: estimated machine input power based on the converter voltage/current measurements.
- `Rs(Ohm) used by FAST`: stator resistance value used by FAST®(which should match the machine value)
- `Lq(mH) used by FAST`: quadrature inductance (as set by the slider) used by FAST®.
- `Cos-daxis`: Cosine of the estimated flux angle generated by FAST®.
- `Sin-daxis`: Sine of the estimated flux angle generated by FAST®.

Note that variables `Cos-daxis`,`Sin-daxis` are used to generate the controller 'd-axis', used to compare in a vector plot against the IPM PM flux vector. A 'linear load module' is also introduced in figure 9.50, which provides a value of 400 Nm at a shaft speed of 1000 rpm.

The 'PSIMCoupler' module introduced earlier (see figure 9.34) has as input the modulation indices generated by the PIL controller module, DC bus voltage and load torque `Tl_me`. The PWM unit of the converter, set to 10 kHz, provides the sampling trigger `sample` for the measured voltages and currents as processed in the modules `Vscaling to IQ30`, `Iscaling` and `DC_bus Scaling`. Outputs of the 'PSIMCoupler' module are as follows (after scaling if needed):

- `u_DC_conv`: measured converter DC bus voltage.
- `uA_`: measured LPF voltage of half-bridge 'A'.
- `uB_`: measured LPF voltage of half-bridge 'B'.
- `uC_`: measured LPF voltage of half-bridge 'C'.
- `iA_`: measured current of half-bridge 'A'.

- `iB_`: measured current of half-bridge 'B'.
- `iC_`: measured current of half-bridge 'C'.
- `v_star`: Voltage of 'starpoint' (a 'Wye' connected machine is always assumed for control purposes) relative to ground.
- `T_me`: shaft torque generated by IRTF module.
- `v_A`: voltage across phase A of the IRTF module (includes the converter switching process)
- `i_alp`: Clarke converted current i_α of the IRTF model.
- `i_beta`: Clarke converted current i_β of the IRTF model.
- `the_me`: shaft angle of IRTF IPM model (used for sensored FOC, if needed).
- `n_m`: shaft speed of IRTF IPM model (used to check operation of FAST®).
- `sample`: Trigger for ADC sampling of measured voltage/currents (can be divided by factor using the `SAmple Divider` module if needed).

Both measured voltage and currents are Forward Clarke converted for use by the PIL controller, as may be observed from figure 9.50. Also given in said figure is the dialogue box 'Converter Variables' as shown in figure 9.51. This

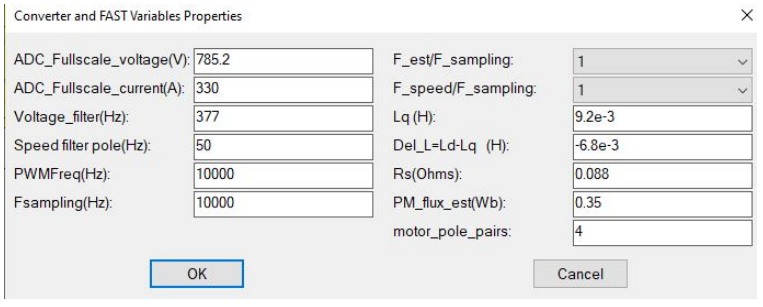


Fig. 9.51 Dialogue box :'Converter and FAST Variables'

dialogue box has the following parameters:

- `ADC_Fullscale_voltage`: A theoretical peak to peak sinusoidal voltage needed to generate a $0 \rightarrow 3.3$V ADC input.
- `ADC_Fullscale_current`: A theoretical peak to peak sinusoidal current needed to generate a $0 \rightarrow 3.3$V ADC input.
- `Voltage_filter (Hz)`: the LPF corner frequency used for removing the PWM component from the converter voltages. Typically, a corner frequency is chosen, that is marginally greater then the maximum (rated) electrical frequency of the machine (but should be a factor 10 lower then the PWM frequency).
- `Speed filter pole (Hz)`: The corner frequency of a LPF filter used by FAST®to measure the shaft speed.
- `PWMFreq(Hz)`: PWM frequency, that must correspond to the value set in the PSIMCoupler unit.

- `Fsampling(Hz)`: Controller sampling frequency, that must correspond to the value set in the PSIMCoupler unit.
- `F_est/F_sampling)`: Basically sets the sampling frequency used by FAST®. If '1' is selected FAST®used the same sample frequency as the controller.
- `Lq(H)`: The direct quadrature inductance of the linear IPM IRTF module. Note, for sensorless control a slider input set the value used by FAST®.
- `Del_L(H)`: The incremental inductance $\Delta L = L_d - L_q$ that defines the direct axis inductance of a linear IPM IRTF module (based on the values of L_q and ΔL). Note, for sensorless control this is not required, as 'option 1' for sensorless control is used (see equation (9.9)).
- `Rs(Ohms)`: Stator resistance of IPM machine, important for low speed operation and must match the value used in the IRTF model.
- `PM_flux_est(Wb)`: A critical parameter for sensorless FOC as this is related to the total flux measured by FAST®using the approach represented by equation (9.9). A judicial choice has to be made, which is dependent on the flux levels that will exceed the PM flux value when running the drive under MTPA conditions. The issue, here is that the scaling limits of the FAST®will be reached (which means that the flux display will be 'frozen' and the speed estimate will have an error) if an incorrect value is set. Hence running the drive over the operational range (under MTPA speed/current control) is needed to check for the best value to be used that covers the flux variation expected. This can also be checked by switching the drive to sensored FOC and observe if FAST®measures the flux and speed (without hitting the internal fixed point limits).
- `motor_pole pairs`: number of motor pole pairs.

Also given in said figure is the dialogue box 'Operational Variables' as shown in figure 9.52.

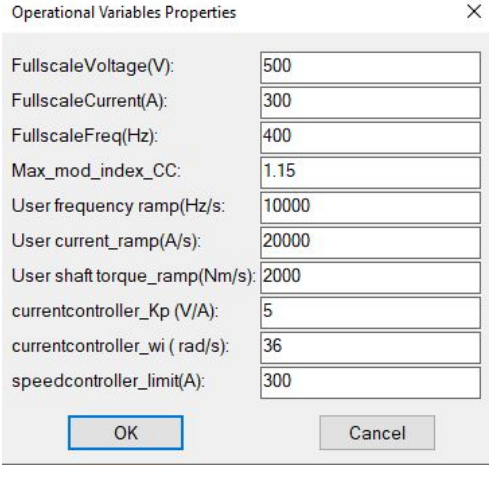


Fig. 9.52 Dialogue box :'Operational Variables'

This dialogue box has the following parameters:

- `FullscaleVoltage(V)`: Scaling factor required to move from fixed point (used in the controller) to actual voltage levels.
- `FullscaleCurrent(A)`: Scaling factor required to move from fixed point (used in the controller) to actual current levels.
- `FullscaleFreq(Hz)`: Scaling factor required to move from fixed point (used in the controller) to actual frequency levels.
- `Max_mod)index_CC`: This is the circular limit of the modulation index vector set in the synchronous current control. The maximum value is $2/\sqrt{3}$, as discussed earlier, see figure 2.38. However, a lower value should be used IF shunts are used in the lower converter switches to measure phase currents (as discussed earlier, see alinea below figure 2.40).
- `User frequency ramp (Hz/s)`: Rate offrequency reference change in the controller.
- `User current ramp (A/s)`: Rate of reference current change in the controller.
- `User shaft torque ramp (Nm/s)`: Rate of shaft torque reference change in the controller.
- `Current controller_Kp(V/A)`: The current controller gain, defined by the quadrature inductance and controller sampling frequency.
- `Current controller_wi(rad/s)`: The current controller bandwidth defined by the ratio of the stator resistance and quadrature inductance (which gives a 'ballpark' figure).
- `Speed controller_limit(A)`: The current vector amplitude limit used by the speed controller.

Moving inside the `IPM-FluxMotor FAST-PIL` reveals the structure shown in figure 9.53. This is the module that needs to be compiled to generate the

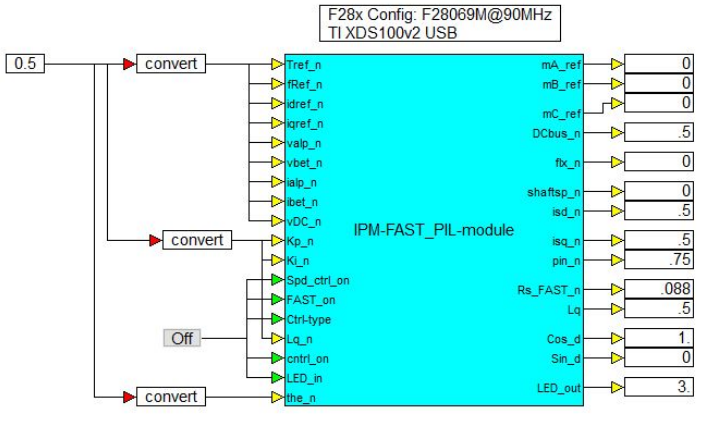


Fig. 9.53 Module:'IPM-FluxMotor-FAST-PIL'

xxx.out file used in the target interface module used by the PIL module shown in figure 9.50. Moving one level into this module leads to the content given in figure 9.54. This shows a LED_ctrl module to drive an LED on the

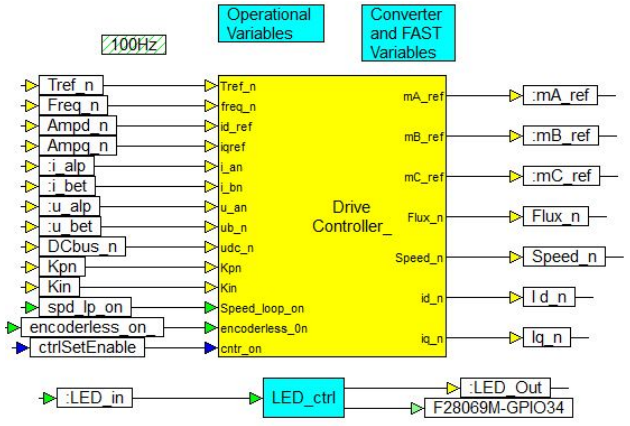


Fig. 9.54 Content PIL module, Field-oriented torque/speed control with FAST®

MCU when PIL operation is active. Also shown are the two dialogue boxes given earlier (see figure 9.50) and the content in both files MUST BE the same. Moving one level into the Drive Controller_ module leads to the content shown in figure 9.55.

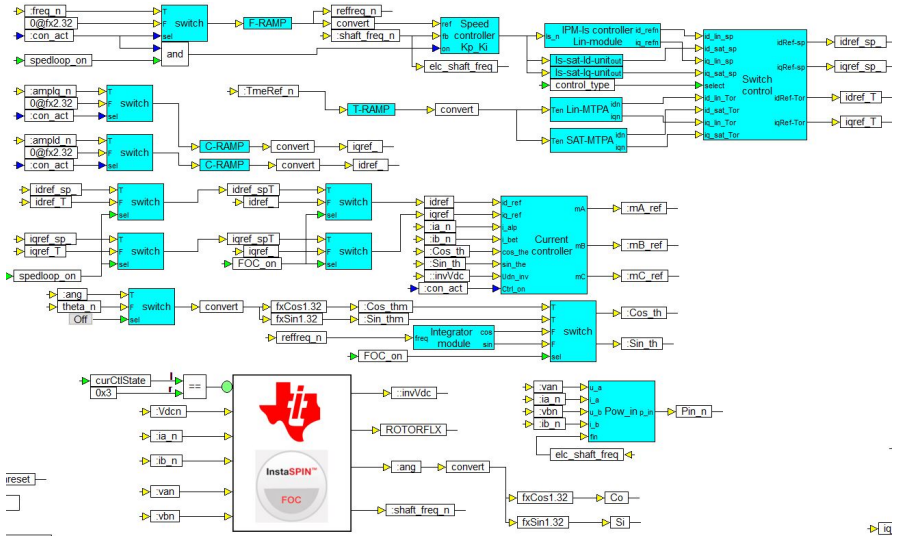


Fig. 9.55 Content IPM 'Drive Controller' module

Readily apparent is the synchronous 'current controller' module, as discussed earlier, see figure 4.125. Both proportional gain, bandwidth and the maximum modulation index of said controller must be provided in the 'operational variables' dialogue box. The speed controller module, as introduced earlier, see section 3.3, must be provided with appropriate proportional and integral gain values, that depend on the combined load inertia and user defined speed control bandwidth (see equation (2.4)). A set of sliders is introduced in figure 9.19 for this purpose.
Output of the speed control unit is the scaled current vector amplitude which is connected to modules `IPM-Is controller-Lin module` and the modules `Is-SAT-Id unit`,`Is-SAT-Iq-unit`. When the controller option 'Linear' is selected, module `IPM-Is controller-Lin module` is active the scaled direct/quadrature currents of the `switch control unit`, are those generated said module.
The content of module `IPM-Is controller-Lin module` is reflected by equations (9.5), (9.2) and the 'linear' curves shown in figure 9.16. When controller option 'Saturation' is selected, the `Switch control` unit takes the inputs from the two modules `Is-SAT-Id unit`,`Is-SAT-Iq-unit` and passed onto variables `idref_sp`, `iqref_sp`. The modules `Is-SAT-Id unit`,`Is-SAT-Iq-unit` represent third order polynomials, as shown by the curves 'sat-case curve fit' in figure 9.16.

When the option 'torque control' is selected the output variables `id_ref_T`, `iq_ref_T` (of the module `Switch control`) are used by the synchronous current controller. The scaled shaft torque, variable `Tme_ref_n` (value set by the `T_ref` slider in figure 9.19) is used by modules `Lin MTPA`,`SAT MTPA`. Module `Lin MTPA` (as discussed earlier, see figure 9.6), will be active when controller option 'Linear' is selected, in which case the outputs are as shown by the 'linear' curves in figure 9.17. If there is a need to represent saturation, when using torque control, controller option 'Saturation' must be used, in which case the outputs of the module `SAT MTPA` are used by the synchronous current controller. The content of the module `SAT MTPA` are third order polynomials represented by the curves 'sat case curve fit' in figure 9.17.

The module `InstaSPIN FOC` contains the FAST®sensorless algorithm and has as inputs the Clarke converted per unit currents and LPF converter voltages. Outputs are the per unit inverse DC bus voltage `invVdc`, estimated flux `ROTORFLX`, orientation of the flux vector `ang` in the α, β reference frame and estimated electrical shaft frequency `shaft freq_n`.
Note that a `Switch` module is used, which enables the user to choose the angle input from either FAST®(use of variable `ang`) or shaft encoder (use of variable `thea_n`). This option is included for debugging sensorless control (by switching to sensored control). This requires recompilation of the PIL module first.

9.2.2.1 Results, IPM sensorless FOC using FAST®

This subsection considers controller operation with control type: `saturation`, which matches operation with the non-linear IPM machine model and operation results for torque and speed control are presented.
An example with speed control active, as given in figure 9.56, shows that around time mark 0.4 s the reference speed (set to 1000 rpm) is reached. Correspondingly, the speed controller provides the direct/quadrature currents (consistent with MTPA operation) needed to generate a shaft torque that matches the load torque. A linear load torque/speed characteristic is assumed, as discussed earlier, which is set to produce 400 Nm at 1000 rpm.

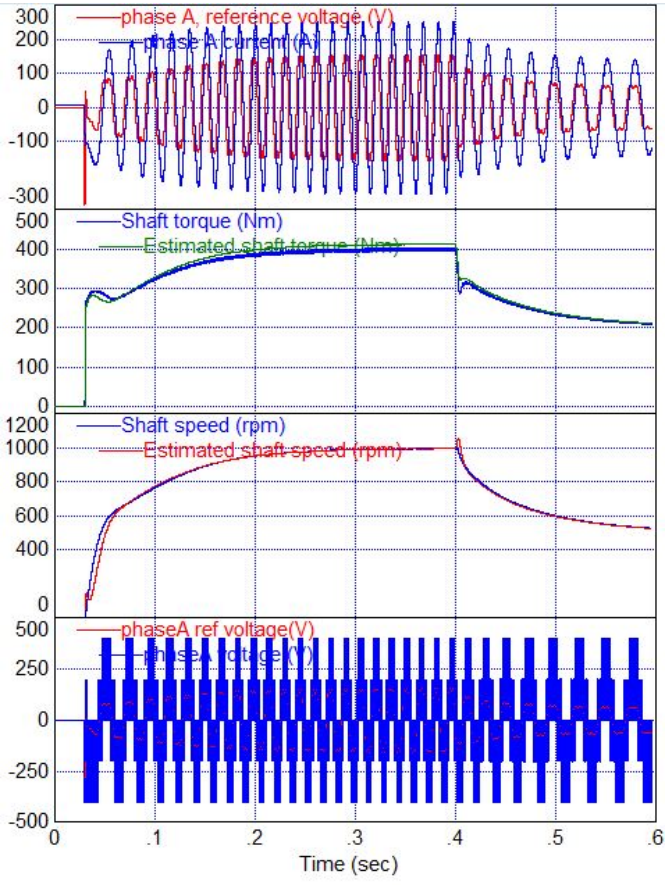
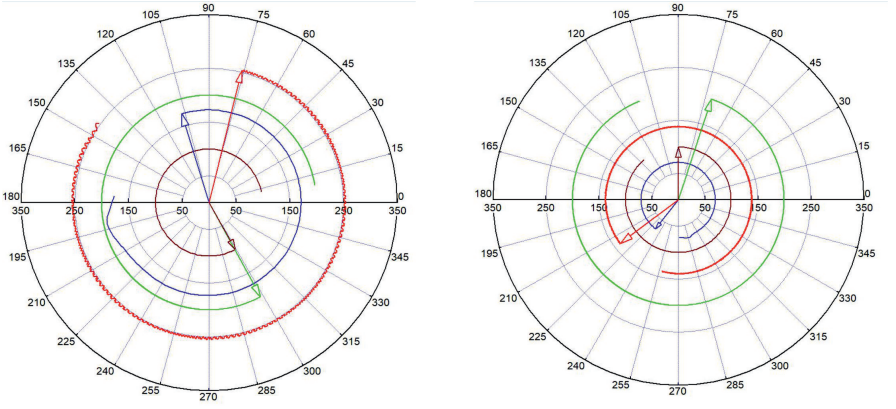


Fig. 9.56 PIL-COSIM result, IPM operation, FOC-speed control active

Added to the plots shown in figure 9.56 is the estimated shaft speed generated with FAST®and the estimated shaft torque calculated using

$T_{\text{me}} = {}^{3}/_{2}\, p\, i_q\, \psi_{\text{tot}}$, where ψ_{tot} is the combined flux due to PM and incremental inductance (see alinea below equation (9.9)). These results confirm that the required speed with matching shaft torque is provided by the drive. At time mark time mark: 0.4 s, the speed reference is changed instantaneously from 1000 → 500rpm. Observation of these results shows that the estimated torque and shaft speed remain accurate. However, the issue of d-axis controller alignment with the PM flux vector changes. This can be observed with the aid of figure 9.57, which shows two vector plots, at time mark: 0.4 s and 0.6 s, that represent operation at 1000 rpm (plot (a)) and 500 rpm (plot(b)) respectively. Both plots show the orientation of (amplitude 100) controller d-axis ('brown'), scaled PM flux vector ('green', amplitude 200), voltage vector ('blue') and the current vector ('red'). Observation of these results learns that controller d-axis deviates from the PM vector when the speed changes from 1000 → 500rpm.



(a) Speed 1000 rpm

(b) Speed 500 rpm

Fig. 9.57 PIL-COSIM result, IPM operation, FOC-speed control active, vector plot

The principle reason for this is that for operation at 1000 rpm the quadrature inductance was tuned to zero the angle between PM vector and controller d-axis, which was achieved with $L_q = 2.148$mH (which is significantly different with the linear model value). When operating at reduced load the current vector amplitude is lower and saturation levels reduce (thus the effective L_q increases in the IRTF machine model), hence the quadrature inductance value used by the controller also needs to be adjusted (via the slider, or lookup table) to achieve PM vector and controller d-axis realignment. For this operating point a quadrature inductance of $L_q = 3.38$mH is required. Hence, in practical terms it is prudent to introduce a current dependant $L_q\,(i_s)$ function, which can be obtained by running the drive under sensored control and zeroing the phase error between PM vector and controller d-axis as vector current amplitude is increased.

The results shown in figure 9.58, represents operation under torque control with T_{ref} =300 Nm, hence the steady state shaft speed is dependant on the applied load torque. Note that at start up ramp functions are used, which determine the rate of reference currents/shaft speed and shaft torque changes, as set in the 'operational Variables' dialogue box.

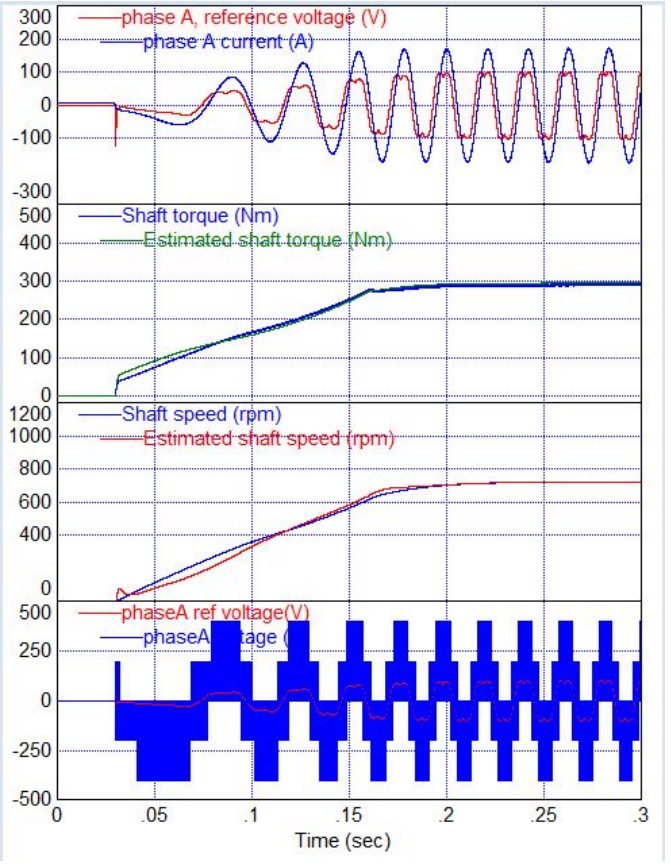


Fig. 9.58 PIL-COSIM result, IPM operation, FOC-torque control active

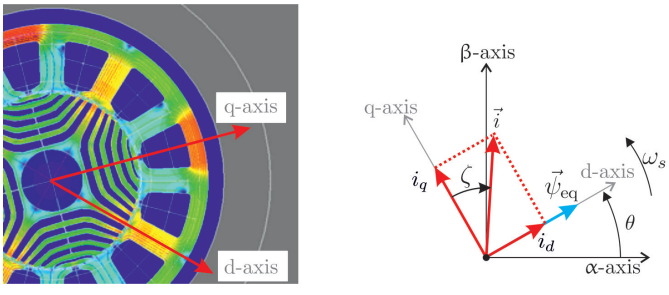
Furthermore, and most importantly, the (controller) quadrature axis inductance was tuned to zero the phase error between PM vector and controller d-axis. This was achieved by setting the controller quadrature inductance slider value to a value of $L_q = 2.52$ mH (after running an example under these load and operational conditions).

The importance of this example should not be under estimated as it highlights the need and use of comprehensive software tools like: FluxMotor®,

PSIM®and Embed®. Furthermore, this example was focussed on sensorless control, but other issues such as: converter switching losses (which can be considered with PSIM®), thermal analysis (which can be considered with FluxMotor®) of the machine should also be looked at. Experimental verification using the HIL approach is also an important step in the developmental tool-chain.

Chapter 10
Synchronous Reluctance (SynRM) drives

'Synchronous Reluctance' machines known as (SynRM) are characterized by a rotor, which has saliency as may be observed from figure 10.1(a). In this context the SynRM can be viewed as an IPM machine without magnets and both use the same stator winding concept. For IPM and SynRM machines the stator inductance is a function of the rotor angle with two extreme cases defined by the (d, q) axis inductance values, as shown in figure 10.1(a). Note that the machine example shown in said figure is two pole pair $p = 2$, and the angle between 'd' and 'q'axes is generally defined as $\pi/2\,p$, which for this machine is equal to 45 deg.



(a) Two pole pair SynRM (b) Flux-current vector plot

Fig. 10.1 Synchronous reluctance topology [24] and flux-current vector plot

The term $(L_d - L_q)$ known as the 'incremental inductance' ΔL, is greater than zero, which is why a positive i_d is required to generate an equivalent flux ψ_{eq}, shown in equation (10.1).

$$T_{\mathrm{e}} = \frac{3}{2}\, i_q\, \psi_{\mathrm{eq}} \tag{10.1a}$$
$$\psi_{\mathrm{eq}} = i_d\, \Delta L \tag{10.1b}$$

Torque production is dictated by a judicial choice of direct and quadrature currents and in this context it is helpful to pursue the 'Maximum Torque per Amp' (MTPA) approach. This issue can be conveniently resolved by determining the optimum angle ζ (shown in figure 10.1(b)), which produces the highest torque for a given current amplitude $|\vec{i}|$.
For the magnetically linear case the process of determining the optimum ζ value that yields the highest shaft torque makes use of expression (10.1), together with a constraint that the current vector locus is a circle with given value i_{s}. as shown in expression (10.2), which also shows the direct/quadrature current values as function of the angle ζ.

$$i_d = i_s \sin\zeta \tag{10.2a}$$
$$i_q = i_s \cos\zeta \tag{10.2b}$$
$$i_s^2 = i_d^2 + i_q^2 \tag{10.2c}$$

Note that the angle ζ is defined as positive (which implies that the current vector is in the first quadrant in the d, q coordinate frame), as shown in figure 10.1(b).
Substitution of (10.2a),(10.2b) into (10.1) gives:

$$T_e = \frac{3}{4} \Delta L\, i_s^2 \sin 2\zeta \tag{10.3}$$

Partial differentiation of the electrodynamic torque with respect to the angle ζ gives:

$$\left(\frac{3}{4} \Delta L\, i_s^2\right) \frac{\partial T_e}{\partial \zeta} = 2\cos 2\zeta \tag{10.4}$$

The optimum angle ζ^{opt} can be found with the aid of (10.4) and using the condition $\frac{\partial T_e}{\partial \zeta} = 0$, which leads to expression:

$$\cos 2\zeta^{\text{opt}} = 0 \tag{10.5}$$

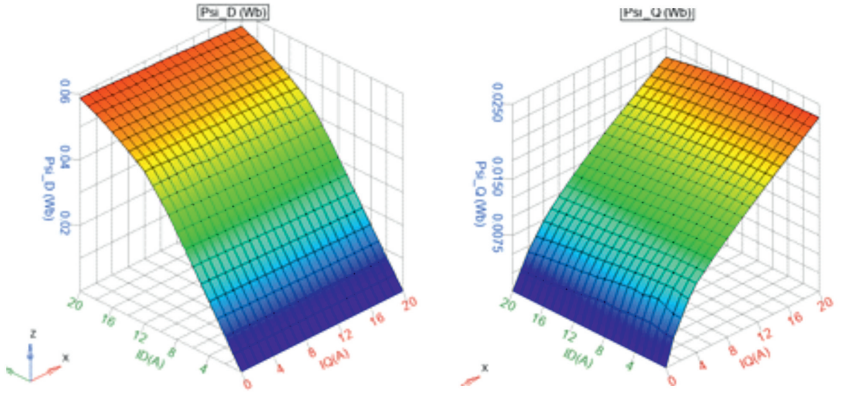
Hence the optimum angle for ζ is simply $\pi/4$(rad). Note that this result is expected, as this analysis is very similar to that shown for a magnetically linear IPM machine (for the case $\psi_{PM} = 0$, no permanent magnets, see equation (9.5)). The reference currents i_d, i_q for the current controller (for a given current amplitude i_s) can now be found using (10.2). Hence the condition defined by expression (10.6) is required to achieve MTPA operation of synchronous reluctance drives, assuming that magnetic saturation is not severe.

$$i_d = |i_q| \tag{10.6}$$

Note that an absolute value of the quadrature current is introduced in equation (10.6), as speed reversals will cause a polarity change and the equivalent flux must remain positive, which implies the need for a positive direct axis current. The issue of magnetic saturation and the need for inverse flux-linkage/current characteristics was mentioned earlier, for IPM machines, see section 2.1.3.3. In these IRTF based machine models a rotor based 'x,y' coordinate system was introduced, which is identical to 'd,q' axes used here. For the synchronous reluctance machine example considered here, a set of flux-linkage/current characteristics was generated [24] (see figure 10.2), which shows the presence of magnetic saturation. The corresponding electrodynamic torque T_e of the SynRM must now be calculated using a generalized torque expression [13], which was also used in IRTF models (see, for example, figure 2.18, which is an SynRM model, if the PM flux is set to zero).

$$T_e = \frac{3}{2}\Im\left\{\vec{\psi}^{*}\,\vec{i_s}\right\} = \frac{3}{2}\left(\psi_d i_q - \psi_q i_d\right) \tag{10.7}$$

Expression (10.7) makes use of direct/quadrature axis flux-linkage variables



(a) Direct axis flux as function of: i_d i_q

(b) Quadrature axis flux as function of: i_d i_q

Fig. 10.2 Flux-linkage/current characteristics of the SynRM [24]

ψ_d, ψ_q, which are a function of the currents i_d, i_q.
For the linear case, the flux-linkages can be found using:

$$\psi_d = L_d i_d \tag{10.8a}$$
$$\psi_q = L_q i_q \tag{10.8b}$$

and substitution of expression (10.8) into (10.7) leads to the linear torque expression shown earlier (see (10.1)), where use is made of the incremental inductance variable $\Delta L = L_d - L_q$. The shaft torque $T_{\text{me}} = pT_e$ of the SynRM for the magnetically linear and non-linear case can therefore be calculated using expression (10.1) and (10.7) respectively, where the latter makes used the flux-linkages/current plots shown in figure 10.2. The results, as given in figure 10.3 shows the shaft torque as function of i_d, i_q for the linear case ('red' grid) and non-linear case, i.e with magnetic saturation.

From a control perspective the aim is to find the 'Maximum Torque Per Amp' (MTPA) for this SynRM, which can be found by plotting the result $T_{\text{me}}/\sqrt{i_d^2+i_q^2}$ (based on the magnetic saturation result shown in figure 10.3), which leads to figure 10.4. Added to figure 10.4 is the optimum ('red' curve) `T/cur` function (which gives the highest shaft torque for the lowest current vector amplitude). These results show that the relationship between the direct/quadrature currents as defined (for linear SynRM models) by expression (10.6) is adequate if magnetic saturation effects are limited.

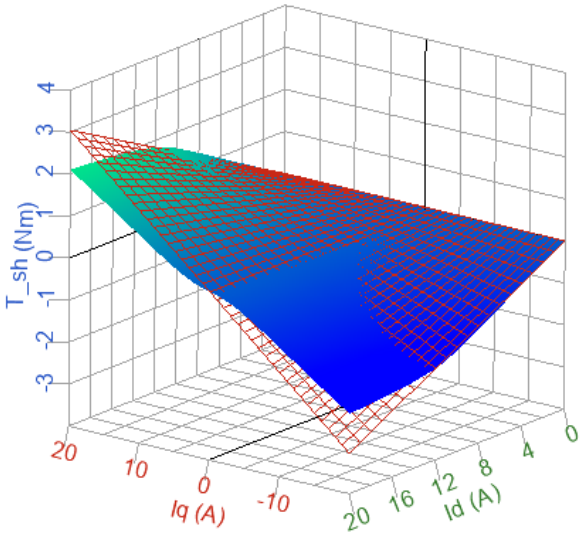


Fig. 10.3 Shaft Torque characteristics of the SynRM with/without magnetic saturation

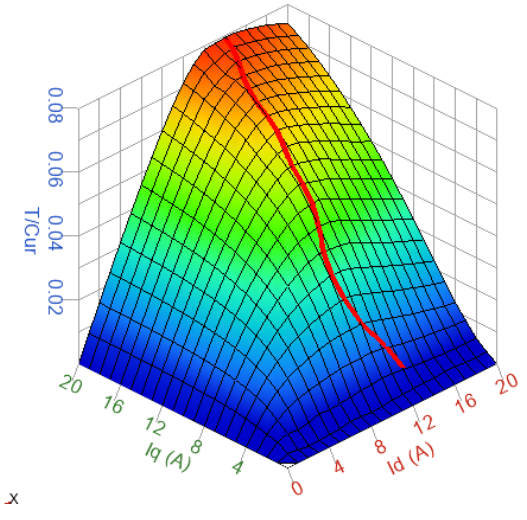


Fig. 10.4 Optimized torque/current (Nm/A) plot as function of i_d, i_q for the SynRM

Observation of these results show the importance (for those cases where linear models are inadequate) of using an SynRM IRTF model with saturation (see figure 2.18, shown for IPM), which raises the issue of finding the inverse current/flux-linkage characteristics of the SynRM. A method to generate these characteristics is shown in figure 9.14. Alternatively, a numerical approach can be used, as developed at RWTH-ISEA [29].

In the ensuing subsections the developmental tool-chain shown in figures 2.43, 2.45 is followed. For the SynRM chapter the toolchain: 'Simulation' (SIM), 'Processor in the Loop'(PIL) and 'Hardware in the Loop'(HIL) will be discussed for a Texas Instruments (TI) based hardware example. The drive control options considered are: Field Oriented Control' (FOC) sensored and sensorless (no encoder). Furthermore, MTPA control is also discussed for a FOC sensored example. With respect to sensorless FOC, a FAST[23] example will be discussed.

10.1 SynRM sensored Field Oriented Control with encoder

The term 'field-oriented control' (FOC) implies that a synchronous 'd,q' coordinate system is tied to the equivalent flux vector of the SynRM machine. More specifically, the controller 'd-axis' is coincidental with the flux vector $\vec{\psi}_{\text{eq}}$ (and machine 'd-axis'), as may be observed from figure 10.1(b). To achieve the best possible torque for the lowest current vector amplitude (referred to as the Maximum Torque Per Ampere (MTPA) operating mode) the condition $i_d^* = |i_q^*|$ must be used (see equation (10.6)).

The term 'sensored' FOC with encoder'(Enc), implies that a shaft encoder is used to provide the angle θ needed to orientate the controller 'd-axis' relative to the 'd-axis' of the SynRM machine. For this type of application, an incremental shaft encoder is often used, which typically. In addition, the shaft encoder produces an index pulse (one pulse/per revolution), which provides the ability to use the incremental encoder for absolute angle sensing purposes (this implies that this function becomes active once an index pulse has been identified).
Note also, that in field-oriented control examples open loop current & frequency control is often used (and added to the examples) as a method to check alignment of the controller 'd-axis' with the equivalent flux vector, when operating open loop (without mechanical load).

10.1.1 SynRM sensored FOC with Enc: SIM

Reference file:

- `SynRM_FOC_enc_SIM.vsm`

This example considers the simulation of a SynRM [24] connected to an 'Average Voltage per Sample' converter, which is in turn connected to a controller, as shown in figure 10.5. The aim is to examine sensored FOC operation of the drive using a fixed point controller, as a prelude to sensorless FOC control using FAST®.

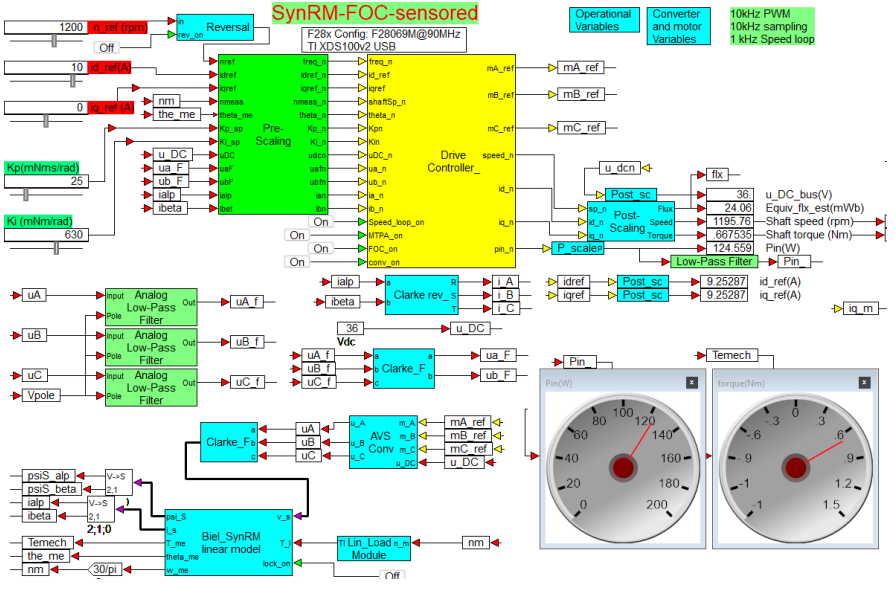


Fig. 10.5 Field-oriented torque/speed control SynRM with encoder: simulation

Inputs to the controller are the direct/quadrature axis reference currents `id_ref`,`iq_ref` and shaft speed reference `n_ref`. Furthermore, two sliders `Kp`, `Ki` are present which set the proportional and integral gain of the speed controller. Four logic inputs are shown that provide the following functionality:

- `conv_on`: converter on/off control.
- `FOC_on`: Field-oriented control active using encoder signal.
- `Speed_loop_on`: Field-oriented speed control active using encoder signal.
- `MTPA_on`: Field-oriented control with Max Torque Per Amp (MTPA) when ON. When OFF the speed controller output is the quadrature current

reference. A'direct-axis current reference' slider controls i_d^* when MTPA is not active.

When the logic inputs `FOC_on` and `Spd_lp_on` are both OFF, drive control is C&F and the direct/quadrature sliders control drive operation together with the shaft speed reference slider.
Also shown in figure 10.5 are three LPF `Analog Low-pass filter` modules that are used in the generic converter to remove the PWM component. Outputs of the forward Clarke converter variables `ualpF`, `ubetF` are used by the controller to calculate motor input power.
Outputs of the controller are three modulation indices, that control the three converter half bridges. The voltages supplied to the motor `uA`, `uB`, `uC` (see figure 10.5) are found by multiplying the modulation indices `mA_ref`, `mB_ref`, `mC_ref` by half the DC bus voltage as executed in module `ConV`, as mentioned earlier. These half-bridge voltages are in effect the 'average voltage per sample t_k' phase variables. The aim of the switching converter is to produce the require average phase voltages for each sample by appropriate converter switching techniques, as discussed previously (see sub chapter 2.1.5). An IRTF based model of the SynRM machine [24] used in this chapter is shown in figure 10.5, and is akin to the generic IPM model shown in figure 2.17 (with $\psi = 0$, no magnets). A linear load module is introduced, which provides a value of 1.0 Nm at a shaft speed of 1800 rpm.

Moving inside the `Drive Controller` reveals the structure shown in figure 10.6. Readily apparent is the synchronous 'current controller' module, as discussed earlier, see figure 4.125. Both proportional gain, bandwidth and the maximum modulation index of said controller must be provided in the 'operational variables' dialogue box. The speed controller module, as introduced earlier (see section 3.3), must be provided with appropriate proportional and integral gain values, that depend on the combined load inertia and user defined speed control bandwidth (see equation (2.4)).
Outputs of the speed control unit is the scaled current vector amplitude when the option `MTPA_on` is selected. Under these conditions the variables `idref_spM`, `iqref_spM` are used as references by the current controller. A saturation unit is used for the `idref_spM` variable to ensure that the minimum direct axis current reference (as defined in the dialog box and set to 5 A) is maintained when quadrature axis currents less then said value occur. If MTPA is not active the output variable `iqref_sp` of the speed controller is used as a reference for the current controller together with the direct axis current variable `idref_tor`, i.e the slider then controls the i_d^* value. If field-oriented control is not active (variable `encoderless_on` zero) operation of the drive reverts to C&F control. A `Pow_in` module with output `Pin_n`, makes use of the Clarke converted per unit measured currents `ia_n`,`ib_n`, Clarke converted LPF voltages `van`,`vbn` and electrical frequency `elec_shaft_freq`.

An example with speed control active, as given in figure 10.7, shows that

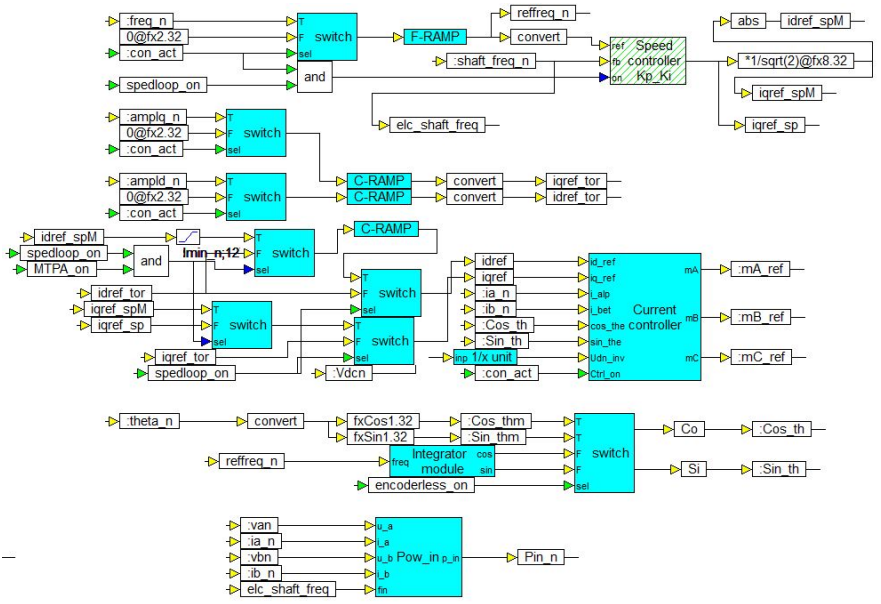


Fig. 10.6 SynRM FOC speed/torque controller structure

around time mark 0.25 s the reference speed (set to 1200 rpm) is reached. For this example MTPA operation commences at time mark 0.318 s. Correspondingly, the speed controller only provides the quadrature current needed to generate a shaft torque that matches the load torque during the period $0 \rightarrow 0.318(s)$. Furthermore, the direct axis current during this initial period is controlled by a slider, set to 10 A). A linear load torque/speed characteristic is assumed, as discussed earlier, which is set to produce 1 Nm at 1800 rpm. At time mark 0.318 s MTPA is activated, which implies that the speed controller now generates a current vector reference i_s^*, which corresponds to $i_d^* = |i_s^*|/\sqrt{2}$, $i_q^* = i_s^*/\sqrt{2}$. Hence, the direct and quadrature axis currents are equal (in absolute sense as defined by condition (10.6)), as may be observed from figure 10.7. Note that the scaled equivalent flux ψ_{eq} shown in figure 10.7 also changes when MTPA mode becomes active, because the direct axis current is then also controlled by the speed module output. These results confirm that the required speed with matching shaft torque is provided by the drive.

Of importance is to consider the vector plot, see figure 10.8 at time mark: 0.6 s, which shows the orientation of the scaled (factor 200) stator flux $\vec{\psi}_s$ ('brown'), scaled (factor 200) unity equivalent flux $\vec{\psi}_{\text{eq}}$ ('green'), voltage vector ('blue') and the current vector ('red').
The relationship between the stator flux generated by the machine and equivalent flux used for control is defined by expression 10.9.

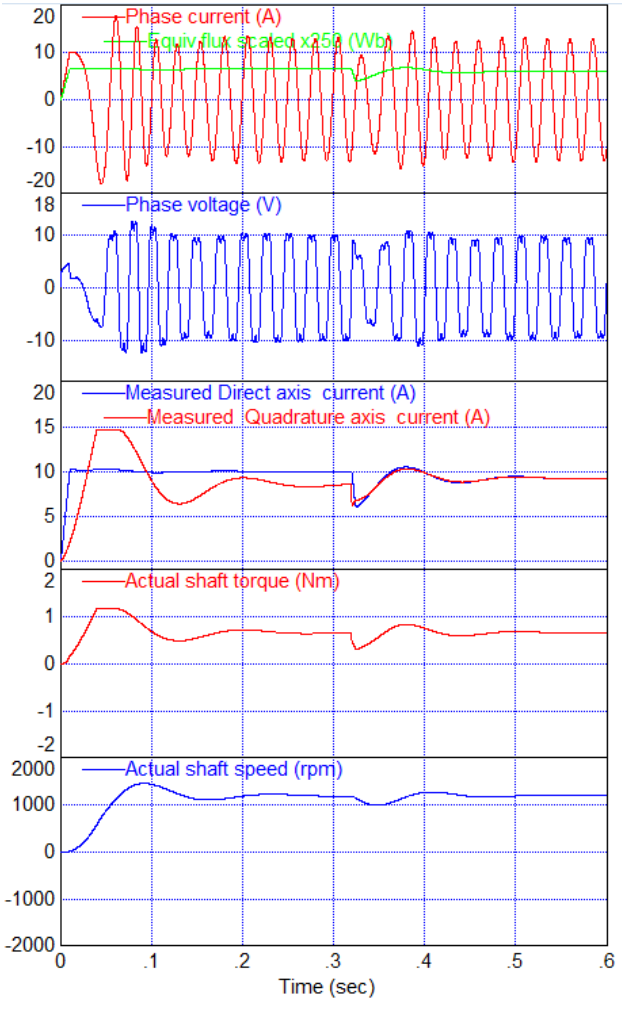


Fig. 10.7 SIM result, SynRM operation, FOC-speed control active: with MTPA active from $t > 0.318(s)$

$$\vec{\psi}_{\text{eq}} = \vec{\psi}_s - \vec{i}_s\, L_q \tag{10.9}$$

The importance of the equivalent flux vector $\vec{\psi}_{\text{eq}}$ should not be under estimated, as it is used to estimate the shaft torque of the machine and for FOC control (d-axis is aligned with said flux vector).

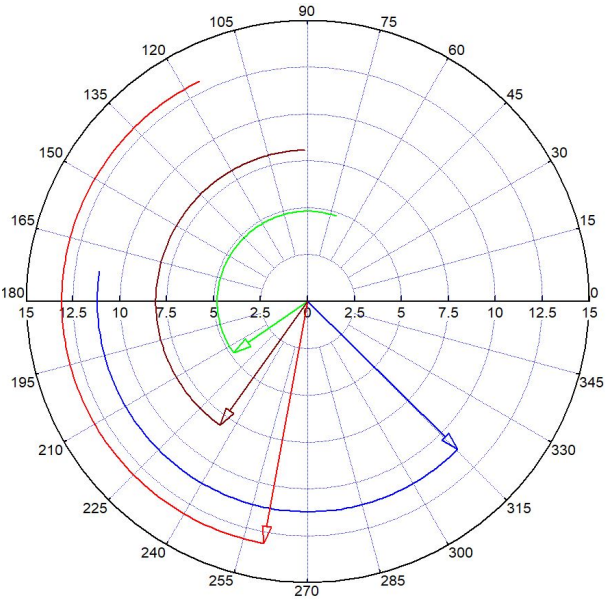


Fig. 10.8 SIM result, SynRM operation, FOC-speed and MTPA control active, vector plot

For sensorless FOC control the EMF vector $\vec{e}_{\text{eq}} = j\omega_s \vec{\psi}_{\text{eq}}$ is used to determine the magnitude and orientation of $\vec{\psi}_{\text{eq}}$ vector in α, β coordinates. The d-axis of the controller is then aligned with said flux vector.

Post Analysis: FOC performance estimation
Reference file:

- `SynRM-FOC-limit-calculator.vsm`, for FOC SynRM drive performance estimation with MTPA active.

The performance of a field-oriented SynRM drive, as considered here is dependant on a number of critical parameters. For example the DC bus voltage and maximum modulation index used set a voltage amplitude locus, which can be used to calculate the maximum EMF that is available. This analysis is based on the use of figure 10.9 and also requires knowledge of the direct/current axis currents and motor parameters. The reference file: `SynRM-FOC-limit-calculator.vsm` calculates the FOC speed limit, with inputs/outputs as shown in figure 8.71 and option 'Max Torque Per Amp' (MTPA) active for this example SynRM[24].
The SIM example shows a SynRM current vector amplitude of 13 A (see figure 10.8) and use of this variable in the FOC performance estimator is shown in figure 10.10.

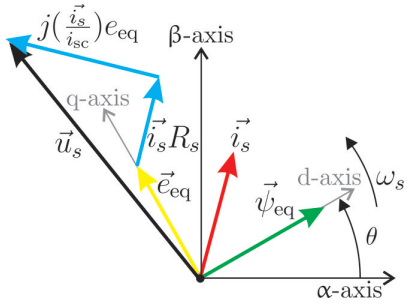


Fig. 10.9 Terminal vector plot with MTPA active

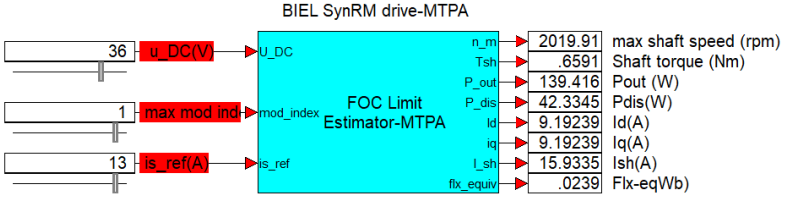


Fig. 10.10 FOC performance estimator of a SynRM drive under MTPA control: operation with $i_s^{\text{ref}} = 13(A)$

These results show reasonable agreement with SIM in terms of the reference direct/quadrature axis currents, equivalent flux level and shaft torque. The FOC speed range for the input variables used, is in this case equal to 2020 rpm. Hence operation of the SIM, with a reference speed of 1200 rpm is well below the limit of FOC operation. Observation of the numerical displays confirms MTPA operation (equal direct and quadrature axis values). Also noticeable is the relatively low equivalent short-circuit current $i_{\text{sc}} = \psi_{\text{eq}}/L_q$ (relative to the current vector $\vec{i}_s$ amplitude) of the example machine, which implies a relatively large voltage across the quadrature inductance as is apparent from figure 10.8. Note that the equivalent short-circuit current can also be expressed as $i_{\text{sc}} = i_d\ (L_d/L_q - 1)$, which shows the importance of maximizing the ratio L_d/L_q of a SynRM machine. For the example machine [24] this ratio is 2.7, which accounts for the large voltage drop across the quadrature axis inductance. This has a negative affect on the power factor, which implies the presence of an unwanted large reactive power component, that reduces machine efficiency.

10.2 Sensorless Field Oriented Control of SynRM

The term 'field-oriented control' (FOC) implies that a synchronous 'd,q' coordinate system is tied to the equivalent flux vector of the synchronous reluctance machine. More specifically, the 'd-axis' of said coordinate system is coincidental with the flux vector $\vec{\psi}_{\text{eq}}$, as may be observed from figure 10.11.

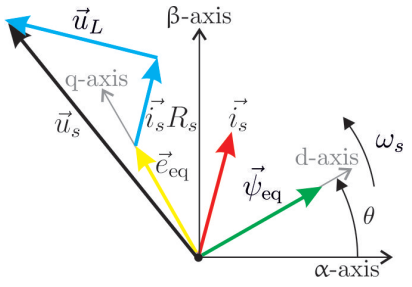


Fig. 10.11 Vector diagram: sensorless field-oriented control (FOC)

The term 'sensorless' FOC, implies the omission of a mechanical rotor position sensor, typically used to align the controller d-axis with the magnetic flux vector. For 'encoderless' (sensorless) field oriented control, use is made of an algorithm, such as FAST®, to identify the magnetic flux vector (amplitude ψ_{eq} and orientation θ in α, β coordinates) of the SynRM. Correspondingly, the 'd-axis' is then aligned with the estimated equivalent magnetic flux vector of said machine.

For SynRM the process of flux estimation is also (as done for IM, PMsyn and IPM) tied to EMF identification, which is based on measurement of the terminal voltage vector $\vec{u}_s$ and current vector $\vec{i}_s$. This EMF identification can be done using 'option 1', as discussed for IPM (see section 9.2). Accordingly the SynRM can be seen as a special case of the IPM, namely with $\psi_{PM} = 0$ (no magnets).
Hence EMF identification, discussed here with aid of figure 10.11 makes use of the voltage $\vec{u}_L = j\omega_s L_q \vec{i}_s$ across the quadrature inductance L_q and voltage across the stator resistance
This approach identifies an EMF due to the equivalent flux $\psi_{\text{eq}} = i_d \Delta L$, with $\Delta L = (L_d - L_q)$ hence, $\vec{e}_{\text{eq}} = j\omega_s \psi_{\text{eq}}$.
Identification of the EMF vector $\vec{e}_{\text{eq}}$ is then carried out using:

$$\vec{e}_{\text{eq}} = \vec{u}_s - \vec{u}_L - R_s \vec{i}_s \tag{10.10}$$

Torque estimation is simplified, as a sensorless algorithm (that uses this EMF estimation approach) identifies the flux ψ_{eq}, which means that a shaft torque estimate can be found using $T_{\text{me}} = {}^{3}/_{2}\, p\, i_q\, \psi_{eq}$.

10.2.1 Sensorless Field Oriented Control of SynRM using FAST®-TI

The example linked to Texas Instruments will be covered using the development tool-chain component: PIL→ HIL based on the use of the Piccolo [23] series and F28069 Launchpad. This MCU has access to the Texas Instruments software InstaSPIN®, which contains their classified sensorless algorithm FAST®. Access to FAST®is by way of an Embed DLL, which implies that simulation is not an option when using this algorithm. Initially the tool-chain component presented in the following subsection will be based on the use of the F28069-DRV8305 drive setup with a SynRM [24].

10.2.1.1 FAST FOC-SynRM-F069-DRV8305:PIL

Reference files:

- `SynRM_FAST_8305_069-PIL.vsm`
- `SynRM_FAST-PIL_v7.vsm`

The Processor In the Loop (PIL) approach discussed in this subsection, is designed to evaluate the controller within the target MCU, in this case the F28069M, which contains the DLL used to communicate with FAST. This implies the use of a target execution module for file, `SynRM-FAST-PIL.out`, as indicated in figure 10.12.
Observation of figure 10.12 shows the presence of an IRTF based linear SynRM module, which is connected to a load module that generates a quadratic torque speed curve, which produces 1.0 Nm at 1800 rpm. An 'average voltage per sample' `AVS conv` module acts as the converter, as discussed earlier (see section 4.1.1.1). The three low-pass filters model the actual R-C filters present in the converter, as used to remove the PWM switching component.
Note that the direct and quadrature axis inductance values used by FAST®(see figure 10.12) are purposely set equal to the quadrature axis inductance value of this machine. The reason for this is that 'option 1', as discussed earlier (see subsection 5.2), is applied here for FOC sensorless control of a Synchronous Reluctance machine.

The target interface module that executes the out-file, generates the modu-

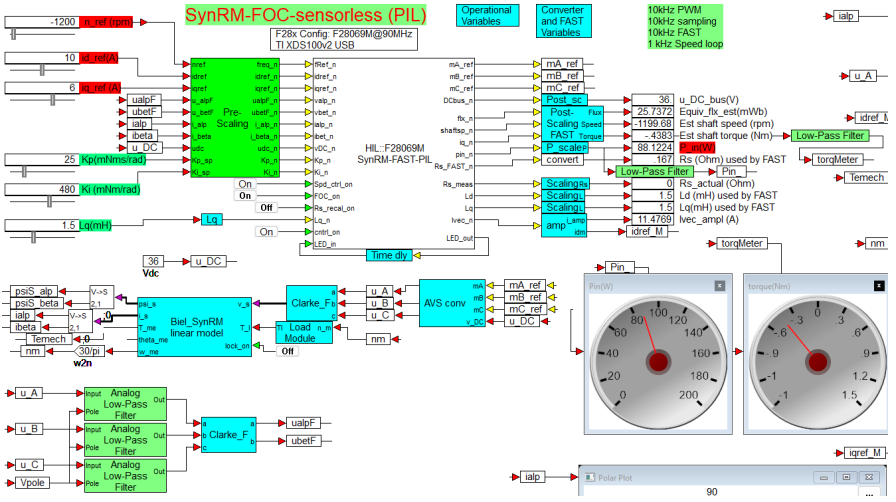


Fig. 10.12 Field-oriented torque/speed control of a SynRM drive using FAST®:PIL operation

lation indices required for the converter/motor, and these components are not affected by use of the PIL. Generation of the out file is based on the PIL module (see figure 10.13), which must be compiled using the command: `toolbox>codegen`.

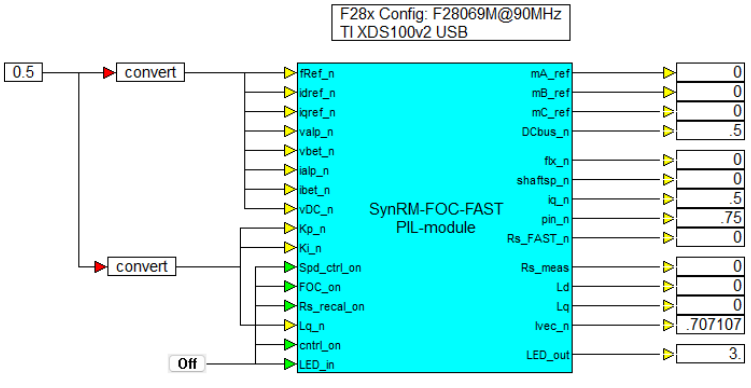


Fig. 10.13 Module:'SynRM-FOC-FAST PIL'

Moving one level into the PIL module `SynRM-FOC-FAST PIL-module` (see figure 10.14) shows a 'Drive Controller'.
Added to said controller is a module `LED_ctrl`, which is used to flash an LED on the MCU board to show that the PIL control is operational.

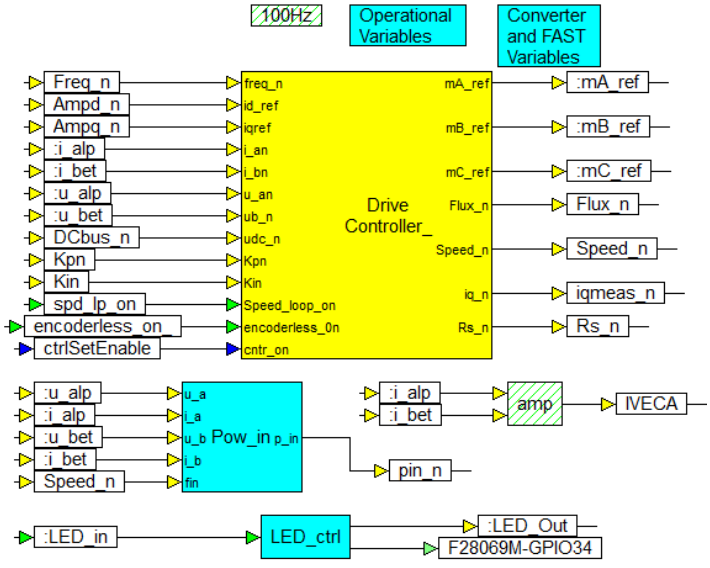


Fig. 10.14 Content PIL module, Field-oriented torque/speed control of a SynRM with FAST®

Furthermore, the modules responsible for input power measurement `Pow_in` and current vector amplitude `amp` are located outside the 'Drive Controller'.

Moving one level down into the `Drive Controller_` module, leads to a set of modules, given in figure 10.15. The `Speed Controller` and `Current controller` shown in figure 10.15, are identical to those introduced earlier, see section 10.1.1.

The current controller requires a per unit inverse bus voltage input, shown as variable `invVdc`, which is now generated by the FAST®module `InstaSPIN FOC`. Said unit provides per unit estimates of the rotorflux amplitude `ROTORFLX`, rotor flux angle `ang`, and electrical frequency `rotfreq`. Inputs to the FAST®module are the per unit measured LPF phase voltages and currents, as mentioned earlier. In addition, a variable `CurCTLstate` (generated in the `100 Hz` module) activates the FAST®module, which is also provided with the per unit measured DC bus voltage `Vdcn`. Also shown in figure 10.15 is a module `Rs_estimation`, which (when activated by the user with variable `Rs_ID`) estimates the stator resistance by injecting a direct axis current (amplitude set in the 'Operational variables' dialogue box). When stator resistance estimation is active the angle output of the FAST®module is held to zero. When FAST®is not used (encoderless off) the logic variable `encoderless_on` (see figure 10.15) is zero and this implies open loop current&frequency control is active. However, the FAST®module will also remain active under these

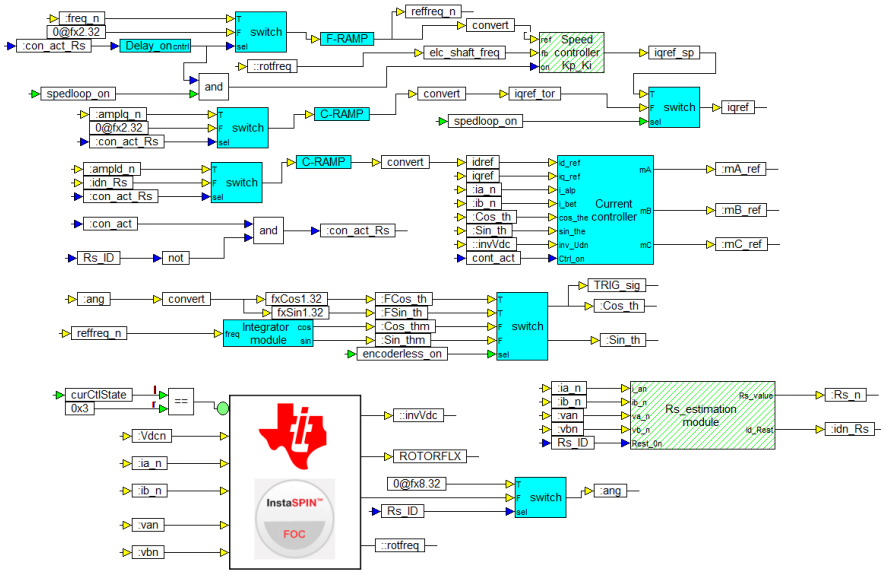


Fig. 10.15 Content 'Drive Controller' module

conditions, hence an effective mode to test operation of this module.

Running the PIL file by starting the file ('green arrow') will show an active 'red' LED on the Launchpad, which indicates that the MCU is operational.

Operation of the drive is set with a reference speed of 1200 rpm and the load controls the quadrature current amplitude (once steady state operation is achieved), as mentioned earlier. FAST® operation starts at 0.3 s and a speed reversal takes place (by changing the speed reference at 0.6 s from 1200 rpm→1200 rpm), as may be observed from figure 10.16.

The vector plot (see figure 10.17), corresponds to time mark 0.6 s (just before the speed reversal), and this is important because it shows the scaled (by a factor 200) equivalent flux vector ('green'), current vector ('red'), and terminal voltage vector ('blue').
The key observation from this vector plot is that angle between the equivalent flux vector and current vector is NOT equal to 45 deg, as was the case for the sensored FOC example (see figure 10.8), operating with the same speed reference. The key difference is that the sensorless example, does not use MTPA operation, hence the direct axis current is held at its rated value of 10.0 A. The reason for this is that with MTPA operation the direct axis current tracks the quadrature current, which is dictated by the shaft torque load.

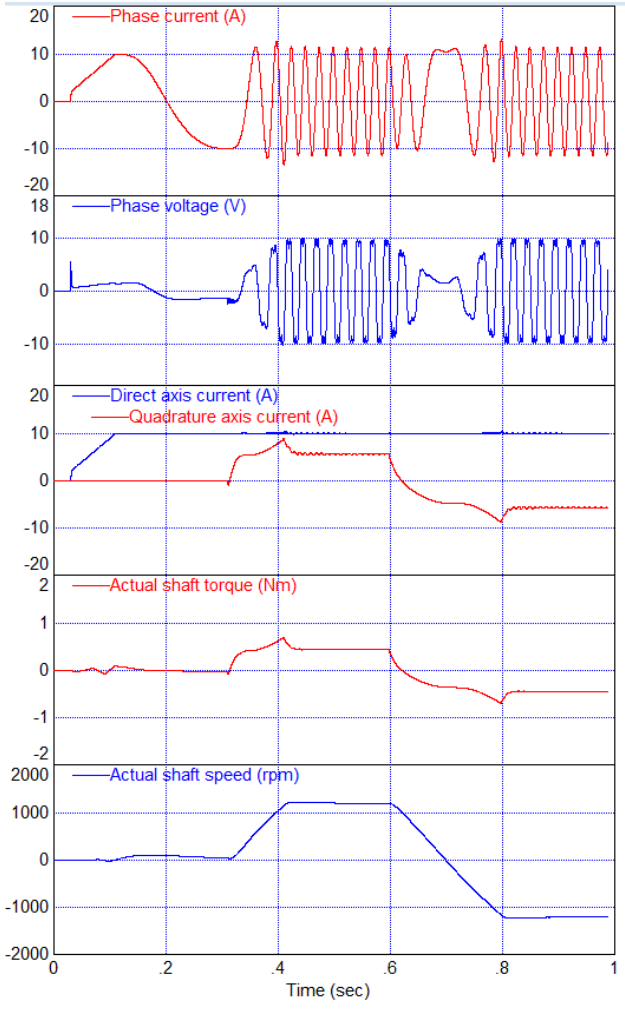


Fig. 10.16 PIL Result: FOC sensorless operation of a SynRM drive using FAST®with speed control

The issue that needs to be considered is that large changes of the direct axis current, cause large changes in the equivalent flux and the FAST®estimator must be able to track these variations. There are limits, to the flux changes that FAST®will and can accept, hence for this example MTPA is disabled.

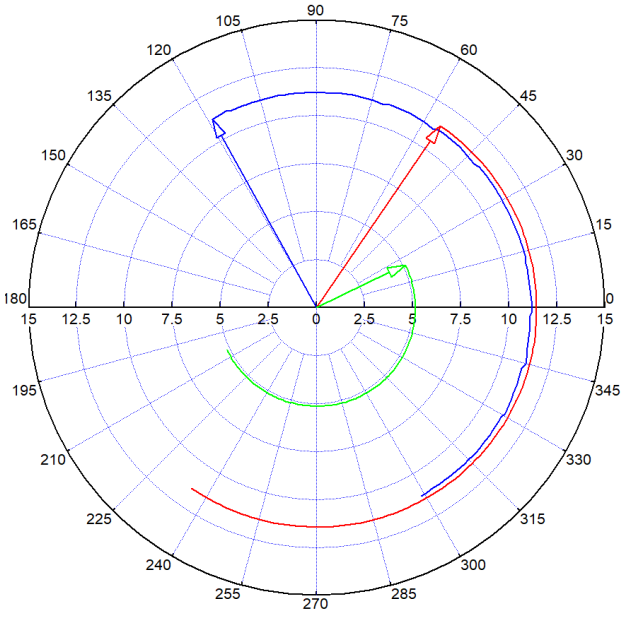


Fig. 10.17 PIL result, SynRM drive operation, sensorless FOC-speed control active: vector plot

10.2.1.2 FAST FOC-SynRM-F069-DRV8305:HIL

Reference files:

- `SynRM_FOC_FAST_8305_069-compile-N.vsm`, for HIL-compile
- `SynRM_FOC_FAST_8305_069-run`, for HIL-debug

The overall drive setup for the 'Hardware In the Loop' (HIL), shown in figure 10.18, represents the `DRV8305` converter (located on the `LAUNCHXL-F28069` module), which is connected to a SynRM machine[24].

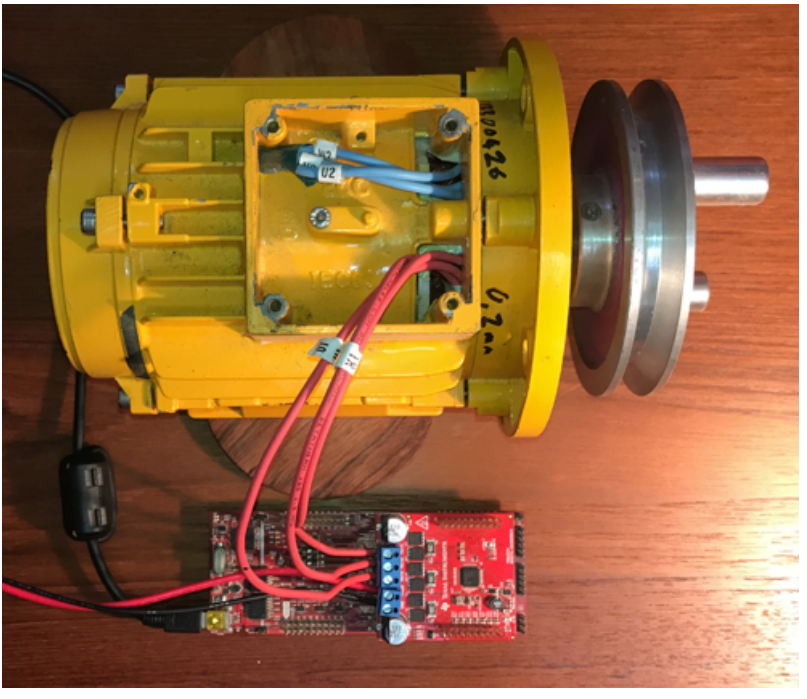


Fig. 10.18 SynRM-DRV8305 drive for HIL operation

For HIL operation, the drive board itself needs to be known in considerable detail. Aspects such as allocation of data acquisition channels relative to the converter voltage/current sensors, choice of PWM outputs and allocation of gate-driver outputs which activate the converter switches.
A compile stage, is used to generate file: `XXX.out` that is used in the 'debug'(run) phase of HIL operation. Hence, the primary task for the HIL-compile stage is to assemble all the modules needed for drive operation with a given controller configuration. The model given in figure 10.19 shows the `FOC_FAST` module, which must be compiled to generate an out-file.

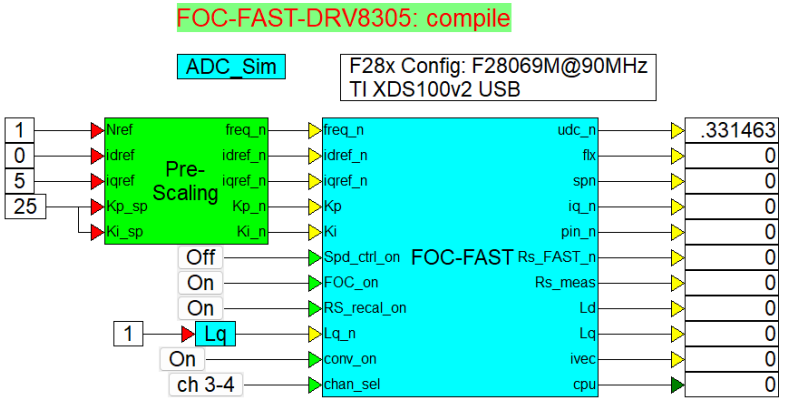


Fig. 10.19 HIL-Compile model of a SynRM drive with field-oriented sensorless control using FAST®

Inputs, shown in this stage as constants (for readability purposes), for this controller module are the user direct and quadrature axis reference currents `idref`, `iqref` and the reference shaft speed `Nref`, which are converted to 'per unit' values in the 'pre-scaling' module. In addition, the proportional and integral speed gain inputs `Kp_sp`, `Ki_sp` are provided to tune the speed controller. Also added to this example is an input for tuning/setting the quadrature axis inductance used by FAST®. Five logic inputs are provided, of which two allow the user to turn on the converter and to select which combination of diagnostic channels should be viewed.
These define the following functions:

- `Sp_ctrl_on`: Speed loop control, when active the shaft speed slider controls the speed, based on the estimated shaft speed generated via FAST®.
- `FOC_on`: Activates sensorless FOC using FAST®. Note when OFF the FAST®algorithm is still operational and this is useful when C&F control is active (both `Sp_ctrl_on` and `FOC_on` buttons: OFF).
- `Rs_recal_on`: Measures the stator resistance at start-up (when `conv_on` is ON) by injection of a direct axis current (value set in the 'operational variables' dialogue box). The duration of the stator resistance estimation time is set in the `100Hz` module via a time delay setting. It is important to verify that the estimation time is sufficient to reach a steady state resistance estimate.
- `conv_on`: Activates the controller.
- `chan_sel`: Determines which two measurements are shown on the plot unit.

Outputs of the `FOC_FAST` module are:

- `udc_n`: The per unit DC bus voltage, which must be scaled by the 'post-scaling' module (in the HIL-debug phase) to provide a reading in Volts.
- `flx`: The flux estimate generated by FAST®, which in this case is the equivalent flux ψ_{eq}, as EMF estimation is based on the use of 'option 1', tied to expression (5.2).
- `spn`: A per unit shaft speed estimate `spn` generated via FAST®.
- `iq_n`: The per unit measured quadrature current, which makes use of the estimated flux, in this case the equivalent flux $\psi_{eq} = i_d\,(L_d - L_q)$, and pole pair number to calculate the shaft torque.
- `pin_n`: A per unit estimate of the motor input power. This variable is also useful as an estimate for the drive input power (but this ignores the converter losses) and can in turn be used to obtain an estimate of the DC bus current using $i_{\mathrm{DC}} \approx {}^{p_{\mathrm{in}}}/_{u_{\mathrm{DC}}}$.
- `Rs_FAST_n`: The per unit phase resistance value used by FAST®.
- `Rs_meas`: The per unit stator resistance value generated by the resistance estimator at start-up (when `Rs_recal_on` is ON). When selected the estimated stator resistance value is transferred to the FAST algorithm on completion of the estimation sequence. When not active the value shown remains zero and is not used by FAST®.
- `Ld`: The direct axis inductance used by FAST®, which is equal to the sum of the quadrature inductance (set by the user slider) and incremental inductance ΔL (set in the dialog box). In this case, use is made op EMF estimation based on 'option 1' (see subsection 5.2), which implies that ΔL must be set to ZERO.
- `Lq`: The quadrature inductance used by FAST®, controlled by the slider input.
- `ivec_n`: The measured per unit current vector amplitude.
- `CPU_activity`: Provides the user with a % reading of MCU activity, relative to the controller sampling time.

Also shown in figure 10.19 is an 'ADC-Sim' module, which is used to provide artificial inputs (as preset in this module) to the ADC channels. Henceforth these values will appear in the controller when this file is run. This approach is useful because it allows the user to check if signal levels within the module are appropriate, i.e as anticipated.

Moving one level down into the `FOC_FAST` module ('right' mouse click, after selecting module) leads to the set of modules and dialogue boxes (see figure 10.20).
Two dialogue boxes are used, which contain the parameters that need to be assigned by the user for HIL-operation. The 'Operational Variables' dialogue box (see figure 10.21), is used to assign full scale values for the voltage, current, frequency and the ramp change for the reference current(s)-frequency. In addition, the speed controller quadrature current (absolute value) has been added, which in effect sets the shaft torque limit for the drive. Furthermore, the speed controller sampling frequency is typically set lower then used by

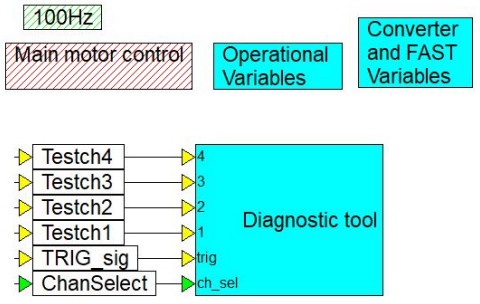


Fig. 10.20 One level down into the FOC-FAST module

the main controller. In this case it is set to 1 kHz. Added to this dialogue box is the variable `Res_est_current`, which is the DC injection current value used for stator resistance estimation when the option `Rs_recal_on` is selected. Note that the modulation index used here, is set to 1.0, which is lower then theoretically possible for three-phase drives. This is due to the use of a converter that makes use of current measuring shunts in the lower converter switches.

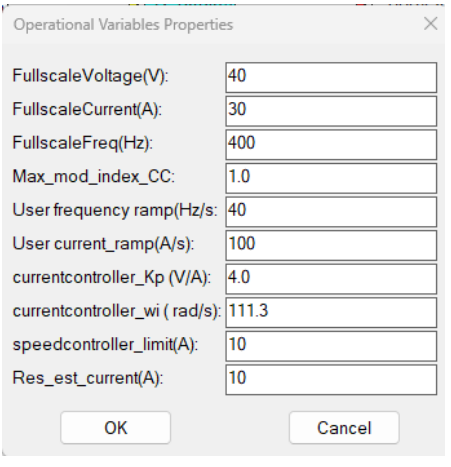


Fig. 10.21 HIL: Dialogue box 'Operational Variables'

The maximum ramp frequency, used for HIL operation has been reduced (compared to PIL) by a factor 10, given the limitations of the power supply unit used for this example. The 'Converter and FAST Variables' dialogue box (see figure 10.22), defines the scaling used by the Analogue to Digital converter.
This module also shows the low-pass filter corner frequency used by the con-

verter for measuring the half-bridge voltages and is an important parameter for FAST®, as it is used to compensate for speed dependent phase angle and amplitude errors of the LPF terminal voltage vector.

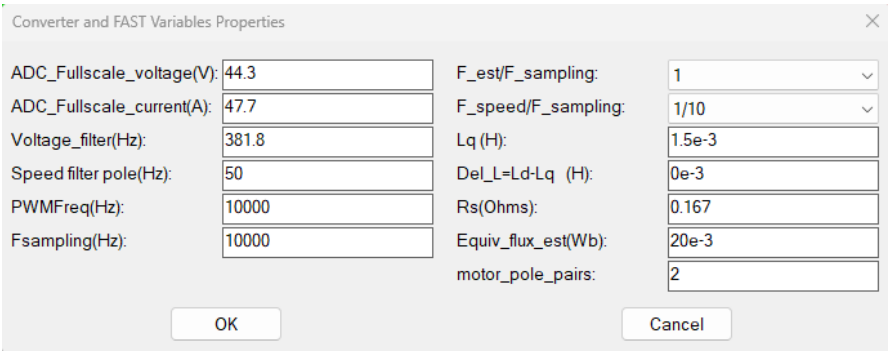


Fig. 10.22 HIL: Dialogue box 'Converter and FAST Variables'

Also added to this dialogue module are a number of FAST®parameters namely:

- `F_est/F_sampling`: set the sampling rate used by FAST®relative to the controller sampling rate, typically set to 1.
- `Equiv_flux)est (Wb)`: an important parameter which represents an estimate for the equivalent rotor flux in Wb. This value can be estimated, by using the drive in the C&F mode, see the 'Post Analysis' 4.2.1.5) and setting the direct axis current value to the rated value, in this case 10 A.
- `Speed filter pole`: corner frequency of the first order LP filter used by the FAST®speed controller.
- `Rs (Ohms)` the stator resistance value which is an important FAST®variable at start-up of the drive.
- `L_q`: the quadrature inductance, which can be estimated using an L-C-R meter (as required to set the current controller gain). The initial value used here, is replaced by the slider value when FAST®is operational.
- `Ld-L_q`: the difference between the direct and quadrature inductance. This value must be set to zero , when using EMF 'option 1' estimation, despite the fact that a SynRM machine is used.
- `p`: pole pair number, which can be found by operating the current in C&F mode and observing the rotational speed of the rotor when, for example setting the reference shaft speed to 60 rpm (one rotation per second).

A 'Diagnostic tool' module is used to buffer two variables, which can subsequently displayed in a plot when operating in HIL-debug mode. A 'two channel' multiplexer is used to allow the user to either display variables, `Test1` and `Test2` or `Test3` and `Test4` by using a 'channel select' button discussed above. This diagnostic tool is extremely valuable for debugging purposes.

A '100Hz' module undertakes all the background tasks for the drive. Among these is the ADC offset control task (state machine used to determine ADC offsets, prior to drive operation). In addition, this module controls a 'heart beat' LED, which flashes on the control card when the MCU is active. Furthermore, this unit generates a number of FAST®settings, which can generally be left unchanged.

The key component shown in figure 10.20 is the `Main motor control` module, which contains the main control algorithm needed for drive control. The unit is shown 'diagonally striped' because the triggering is controlled by the 'end of conversion' (EOC) generated by the ADC unit.
Moving one level lower into the 'Main motor Control' module reveals a set of modules (see figure 10.23). Of these, the 'Drive controller' has been discussed earlier (see figure 10.15), where it was used in phase: PIL of the development tool-chain.

The per unit variables: DC bus voltage, half-bridge currents and voltages are generated by a board specific 'ADC-PWM' module.
This module is configured with the following basic settings :

- PWM frequency: 10 kHz
- ADC sampling rate: 10 kHz
- Converter dead-time setting: 0.1 μs
- Board in use: `LAUNCHXL-F28069` with `BOOSTXL-8305` converter

This ADC-PWM module conforms with MCU timing, as discussed in figure 2.40, which implies that the `Main motor control` module is executed after each EOC pulse, i.e after the ADC sampling and data acquisition process has been completed.
The `ADC-PWM-LaunchXL-069M-8305_aft` unit is used generate the PWM signals for the converter. In addition it is used to measure the three low-pass filtered (per unit) half-bridge voltages and these are then converted to stationary coordinates α, β voltages: `u_alp`, `u_bet`. Three per unit phase currents are measured and Forward Clarke converted to stationary α, β coordinates, with outputs `i_alp`, `i_bet` respectively.
ADC Offsets in this unit, are removed with the aid of a low-pass filter, with an `ADC_offset_freq` corner frequency of 1 Hz (set in the `ADC-PWM` module) and controlled by said module, as mentioned previously. The per unit DC bus voltage: `DCbus_n`, completes the ADC measurement sequence.

The HIL-debug/run tool-chain component is the operational component of this example (see figure 10.24), and its task is to execute the `XXX.out` file compiled in 'HIL-compile' phase (see above).
Central to this figure is the Embed 'target execution module' , which executes said out-file. Three sliders are used to set the direct axis/quadrature axis

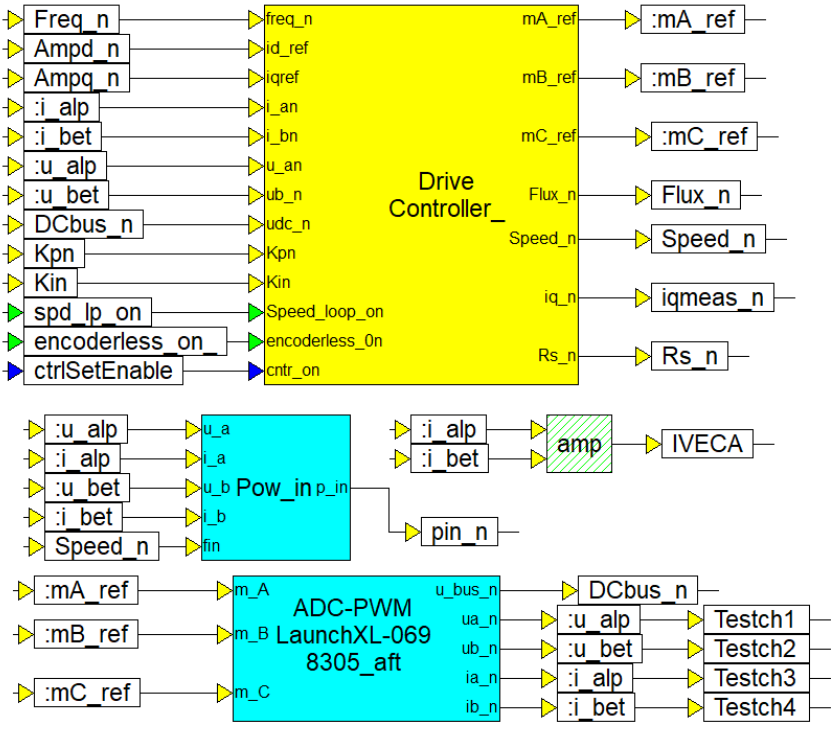


Fig. 10.23 One level down into the 'Main motor controller' module

current reference values and the shaft speed. Three additional sliders are present, of which two are used to se the proportional and integral gain of the speed controller. The third slider, is used to set the quadrature axis inductance value, as required for FAST®.

Post-scaling modules are used to convert the per-unit measured DC bus voltage and % CPU time to actual values, as shown with numeric displays. Furthermore, a set of numerical display units have been added which show: estimated equivalent flux, shaft speed, shaft torque, stator resistance used by FAST®, measured stator resistance at startup (when this function is actively used), direct and quadrature axis inductance values used by FAST®and current vector amplitude. In addition, a set of gauges have been added which show (low-pass filtered): shaft speed, shaft torque, input power and estimated DC supply current.

Two 'Monitor Buffer' modules are used to display two selected (using a button connected to the `Ch_sel` input) diagnostic signals.

The plot given in figure 10.24 displays the per unit measured current vector locus. This requires the channel select button to be set to `ch 3-4`. Selecting `ch 1-2`, will display the per unit voltage vector locus. The actual amplitude

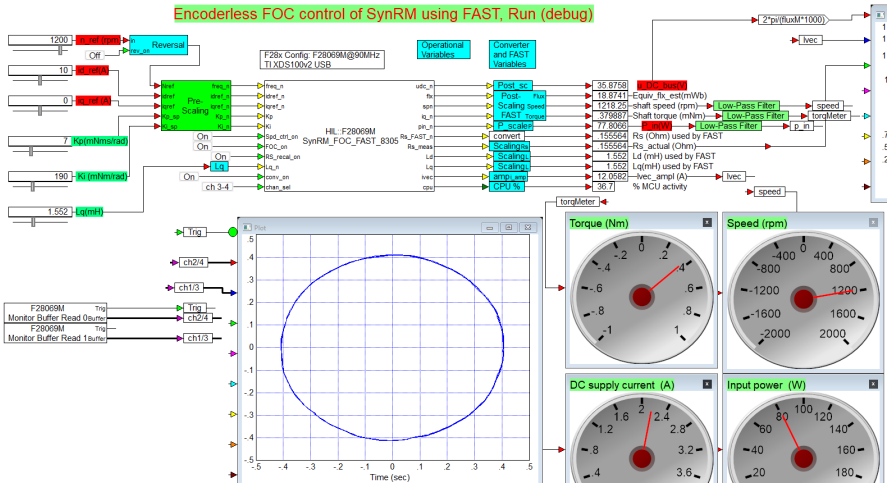


Fig. 10.24 HIL-Debug model, F28069M-DRV8305 of a SynRM drive with field-oriented control using FAST®

of the current locus is found by using the 'Read Coordinates' function of the plot module, which shows a value of 0.41 , that must be multiplied by the 30 A full scale current, which gives a value of 12.3 A. The shaft speed reference shown in figure 10.24 was set to 1200 rpm, which corresponds to an electrical frequency f_s of 40 Hz. These results show operation under encoderless FOC-speed control with the machine operating with a 'hand applied' load condition.

In addition, a DC current probe with 'PicoScope' [20] was used to record the phase current when using the speed reference reversal option (see figure 10.24), which causes a reference speed change from $1200 \rightarrow -1200$ rpm. Such a reference speed step change will cause a current change similar to the PIL result (see figure 10.16) shown earlier, as may be observed from figure 10.25.
This result confirms the versatility of sensorless FOC control, that can be achieved with a SynRM drive.

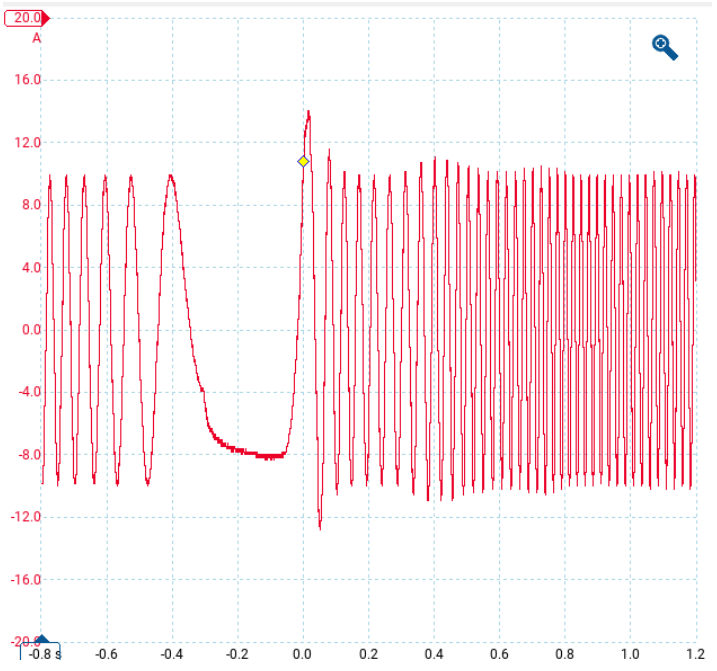


Fig. 10.25 Measured phase current current, F069-DRV8305 drive of a SynRM drive with field-oriented control using FAST®: speed 1200 → −1200 rpm reversal

Prior to activating this type of laboratory, the reader needs to be aware of the fact that he or she is about to activate a complex electrical system, with live voltages. Hence it is prudent, **always**, to execute the following 'Pre-Drive' check list:

- Dialogue boxes used in debug mode, match those used in HIL-compile: the run version is compiled with the dialogue boxes entry specified under HIL-compile. The dialogue boxes shown in HIL-debug are used by the 'Pre/Post' scaling modules.
- Ensure that the sample time used is correct and latest (and correct) out-file has been down loaded to the MCU (Right mouse click on MCU module to show dialogue box of these variables).
- Confirm that the user input values are set to either zero, or 'acceptable' values, which will not cause a current trip of the converter
- Confirm that the converter 'switch' is set to `off` and the power supply is on (DC bus voltage present).
- Confirm that the motor is connected firmly and properly.

After completion of the Pre-Drive check-list, activate the program and confirm that the supply voltage source level shown in the digital display is 36 V. If not stop the program and restart. With the correct DC voltage level established, turn on the converter (using the `on` button, shown) and monitor the motor shaft and diagnostic scope.

Post Analysis: FOC performance estimator

Reference file:

- `SynRM-FOC-limit-calculator-o`, for FOC performance estimation.

The performance of a field-oriented drive, such as the SynRM drive considered here, is dependent on a number of critical parameters. For example the DC bus voltage and maximum modulation index used set a voltage amplitude locus, which can be used to calculate the maximum EMF that is available. This analysis is based on the use of figure 10.11 and also requires knowledge of the direct/current current setting and motor parameters. The reference file calculates the FOC speed limit, with inputs/outputs as shown in figure 10.26.

The dialog box for this drive is identical to the one introduced earlier (see figure 10.10) and closely matches the experimental machine used for HIL operation. The slider setting match what is used for operation HIL and the

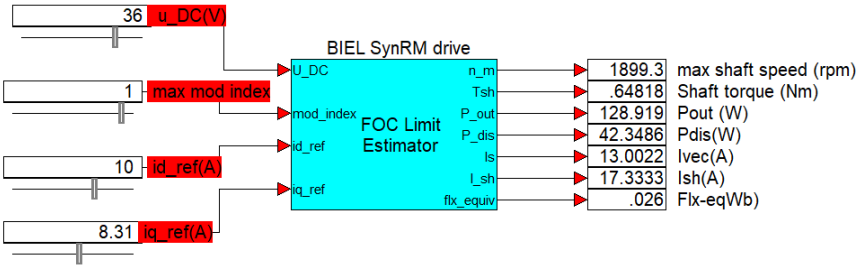


Fig. 10.26 FOC performance estimator of a SynRM drive:operation with $i_q^{\text{ref}} = 8.31(A)$, $i_d^{\text{ref}} = 10(A)$

result show the maximum shaft speed that can be achieved for a quadrature axis current that is set to generate a current vector amplitude of 13 A that matches the value used for MTPA operation (see figure 10.10).

References

[1] Dietz-motoren,'H3-Typ-PEDR80-90/2', *Nameplate data: 0.75 kW, 3000/min, 11.5A, 50V, 50 Hz.*

[2] Technical Brochure, 'Hudson Brushless DC servo motors', *model:PM M-2310P-LN-04K*, Version 2.0, April 15, 2015.

[3] 'Evaluation of the 2004 Toyota Prius Hybrid Electric Drive System', FY2006, US department of Energy.

[4] 'Equipmake', `https://www.equipmake.co.uk/`.

[5] EMOD, 'Type HEFIE2 80L/4A', *Nameplate data: 0.75 kW, 1425/min, 25A, 30V star, 50 Hz.*

[6] ROTOTECH, 'Type Hu 71 B 4 B3', *Nameplate data: 0.37 kW, 1330/min, 1.0A, 415V star, 50 Hz.*

[7] POPE Electric Motors, 'Frame size 63', *Nameplate data: 0.18 kW, 2710/min, 1.32A, 240V, 50 Hz.*

[8] Technical Brochure, 'T-motor', *model: MN3508 KV380-Navigator Type*, `https://store.tmotor.com/`.

[9] Technical Brochure, 'STEPPERONLINE', *model: 17HS24-2104S*, `www.omc-stepperonline.com`.

[10] Technical Brochure, 'Turnigy', *model: Global motors 5208*, `https://rcsearch.info/hobbyking/i45369/`.

[11] Hybrid stepper motor, 'OUKEDA MOTOR', *model: 17HS3417*, `https://fluxworkshop.com/products/afaa100201-ok42sth22-0804ac-nema17-silver`

[12] A. Veltman, D.W.J. Pulle and R. W. De Doncker, 'Fundamentals of Electrical Drives' 2nd edition, *Springer ISBN 978-3-319-29408-7*, 2016.

[13] R. W. De Doncker, D.W.J. Pulle and A. Veltman, 'Advanced Electrical Drives' 2nd edition, *Springer ISBN 978-3-030-29976-2*, 2020.

[14] D.W.J. Pulle, P. Darnell and A. Veltman, 'Applied Control of Electrical Drives', *Springer ISBN 978-3-319-20042-2*, 2015.

[15] Altair Embed,'Visual Environment for Embedded system development', `http://uk.altair.com/embed/`.

[16] Altair PSIM, `https://www.altair.com/resourcelibrary/`.

[17] Altair ACTIVATE, `https://www.altair.com/resourcelibrary/`.

[18] Altair COMPOSE, https://www.altair.com/resourcelibrary/.
[19] Altair FLUXMOTOR, https://www.altair.com/fluxmotor/.
[20] PicoScope6, https://www.picotech.com/products/oscilloscope.
[21] Texas Instruments, 'UniFlash',https://www.ti.com/tool/UNIFLASH.
[22] Texas Instruments, 'High Voltage Motor Control PFC developers kit', https://www.ti.com/tool/TMDSHVMTRPFCKIT.
[23] Texas Instruments, 'INSTASPIN-BLDC solution', https://www.ti.com/tool/INSTASPIN-BLDC.
[24] 'SynRM design for fractional horsepower drives', IEM, Bern University of Applied Science, 2019.
[25] SCIAMBLE 'End to End solutions', https://sciamble.com/.
[26] Trampa 'VESC MRKIII', https://trampaboards.com/vesc-6-mkvi/.
[27] H. Polinder, M.J. Hoeijmakers, M. Scuotto,'Eddy-Current Losses in the Solid Back-Iron of PM Machines for different Concentrated Fractional Pitch Windings' *Delft University of Technology*, IEEE 2007.
[28] A. Hughes, 'Electric Motors and Drives', *Newnes*,1994.
[29] M. Boesing, 'Noise and Vibration Synthesis based on Force Response Superposition', *RWTH-ISEA, Aachen, Germany*, Ph.D thesis, 2013.
[30] Texas Instruments 'UniFlash', https://software-dl.ti.com/ccs/esd/uniflash/docs/v7
[31] BULL RUNNING BR2804-1700KV 'rc motors aeroplane model brushless-Green',
[32] A. Veltman, 'The fish method: interaction between AC-machines and switching power converters ', *Delft University Press*, Delft, The Netherlands, Ph.D thesis, 1993.
[33] A. Simpkins and E. Todorov, 'Position Estimation and Control of Compact BLDC Motors based in Analog Linear Hall Effect Sensors ', *2010 American Control Conference*, Baltimore,MD, USA, 2010.
[34] ST Micro, 'B-G473E-ZEST1S Discovery kit', https://www.st.com/en/evaluation-tools/b-g473e-zest1s.html.
[35] ST Micro, 'STEVAL-LVLP01 evaluation board', https://www.st.com/en/evaluation-tools/steval-lvlp01.html.
[36] ST Micro, 'B-ZEST-ADAPT1 adapter board', https://www.st.com/en/evaluation-tools/b-zest-adapt1.html.